FINAL REPORT
FOR
DIGITAL RECEIVER STUDY
AND IMPLEMENTATION
1 JUNE 1971 – 1 JUNE 1972

CONTRACT NO.: NAS5-11424

PREPARED BY
MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
St. Louis, Missouri 63166
FINAL REPORT
FOR
DIGITAL RECEIVER STUDY
AND IMPLEMENTATION
1 JUNE 1971 – 1 JUNE 1972

CONTRACT NO.: NAS5-14424

GODDARD SPACE FLIGHT CENTER

CONTRACTING OFFICER : K.G. WILLIAMS CODE 245
TECHNICAL MONITOR: W. ALFORD CODE 563

PREPARED BY
MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
St. Louis, Missouri 63166

MCDONNELL DOUGLAS CORPORATION

D.A. FOGLE, DR. G.M. LEE, J.C. MASSEY

FOR
GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND
ABSTRACT

Computer software is developed which makes it possible to use any general purpose computer with A/D conversion capability as a PSK receiver for low data rate telemetry processing. Carrier tracking, bit synchronization, and matched filter detection are all performed digitally. To aid in the implementation of optimum computer processors, a study of general digital processing techniques was performed which emphasized various techniques for digitizing general analog systems. In particular, the phase-locked loop was extensively analyzed as a typical non-linear communication element. Bayesian estimation techniques for PSK demodulation were studied. A hardware implementation of the digital Costas loop was developed.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>v</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. DEVELOPMENT OF DIGITIZED PHASE-LOCKED LOOP</td>
<td>3</td>
</tr>
<tr>
<td>2.1 Synthesis of Equations</td>
<td>3</td>
</tr>
<tr>
<td>2.1.1 Choice of Phase-Locked Loop Filter</td>
<td>3</td>
</tr>
<tr>
<td>2.1.2 Analysis of Linear Loop</td>
<td>4</td>
</tr>
<tr>
<td>2.1.3 Differential Equation Approach</td>
<td>6</td>
</tr>
<tr>
<td>2.1.4 Difference Equation Formulation</td>
<td>9</td>
</tr>
<tr>
<td>2.2 Development of Computer Programs</td>
<td>12</td>
</tr>
<tr>
<td>2.3 Comparison of Techniques</td>
<td>21</td>
</tr>
<tr>
<td>2.3.1 Non-Linear Phase-Locked Loop</td>
<td>21</td>
</tr>
<tr>
<td>2.3.2 Linear Filters</td>
<td>25</td>
</tr>
<tr>
<td>2.4 Determination of System Constraints</td>
<td>30</td>
</tr>
<tr>
<td>2.4.1 Sampling Rate</td>
<td>30</td>
</tr>
<tr>
<td>2.4.2 Quantization</td>
<td>33</td>
</tr>
<tr>
<td>2.4.3 Scale Factor</td>
<td>35</td>
</tr>
<tr>
<td>2.5 Automatic Gain Control (AGC)</td>
<td>40</td>
</tr>
<tr>
<td>2.5.1 Bandpass Limiter</td>
<td>40</td>
</tr>
<tr>
<td>2.5.2 Closed Loop AGC</td>
<td>43</td>
</tr>
<tr>
<td>2.5.3 Comparison of Techniques</td>
<td>48</td>
</tr>
<tr>
<td>2.6 Acquisition</td>
<td>50</td>
</tr>
<tr>
<td>2.6.1 Swept Frequency Method</td>
<td>50</td>
</tr>
<tr>
<td>2.6.2 Fast Fourier Transform Acquisition</td>
<td>52</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.6.3</td>
<td>Free Running Acquisition</td>
<td>57</td>
</tr>
<tr>
<td>2.6.4</td>
<td>Comparison of Techniques</td>
<td>57</td>
</tr>
<tr>
<td>2.7</td>
<td>Hybrid Simulation</td>
<td>61</td>
</tr>
<tr>
<td>2.7.1</td>
<td>Input Filter</td>
<td>61</td>
</tr>
<tr>
<td>2.7.2</td>
<td>Results</td>
<td>66</td>
</tr>
<tr>
<td>2.8</td>
<td>Evaluation of Performance</td>
<td>70</td>
</tr>
<tr>
<td>2.8.1</td>
<td>Phase Error Variance</td>
<td>71</td>
</tr>
<tr>
<td>2.8.2</td>
<td>Phase-Locked Loop Threshold</td>
<td>73</td>
</tr>
<tr>
<td>2.8.3</td>
<td>Acquisition</td>
<td>75</td>
</tr>
<tr>
<td>3.1</td>
<td>Synthesis of PSK Receiver Software</td>
<td>79</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Sampling and A/D Conversion of Data</td>
<td>79</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Rejection of DC Bias</td>
<td>84</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Input Filter Design</td>
<td>86</td>
</tr>
<tr>
<td>3.1.4</td>
<td>Generation of PSK Modulated Noisy Carrier</td>
<td>88</td>
</tr>
<tr>
<td>3.1.5</td>
<td>Study of Carrier Synchronization</td>
<td>89</td>
</tr>
<tr>
<td>3.1.6</td>
<td>Automatic Gain Control</td>
<td>100</td>
</tr>
<tr>
<td>3.1.7</td>
<td>Study of Bit Synchronization</td>
<td>103</td>
</tr>
<tr>
<td>3.1.8</td>
<td>Development of Integrated Program</td>
<td>137</td>
</tr>
<tr>
<td>3.1.9</td>
<td>Study of Acquisition Techniques</td>
<td>147</td>
</tr>
<tr>
<td>3.2</td>
<td>Digital PSK System Analysis</td>
<td>156</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Sampling Losses</td>
<td>157</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Losses Due to Imperfect Phase Synchronization</td>
<td>164</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Losses Due to Imperfect Bit Synchronization</td>
<td>167</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Performance Evaluation</td>
<td>171</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td>STUDY OF NON-ANALOG SYNTHESIS TECHNIQUES</td>
<td>179</td>
</tr>
<tr>
<td>4.1</td>
<td>Bayesian Estimation</td>
<td>179</td>
</tr>
<tr>
<td>4.2</td>
<td>Performance Evaluation</td>
<td>186</td>
</tr>
<tr>
<td>5.</td>
<td>DEVELOPMENT OF HARDWARE COSTAS LOOP</td>
<td>193</td>
</tr>
<tr>
<td>5.1</td>
<td>Machine Organizations</td>
<td>195</td>
</tr>
<tr>
<td>5.2</td>
<td>Parallel, 3 Arithmetic Logic Unit Design</td>
<td>195</td>
</tr>
<tr>
<td>5.2.1</td>
<td>General Description</td>
<td>197</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Function Block</td>
<td>198</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Logic Diagrams</td>
<td>202</td>
</tr>
<tr>
<td>5.2.4</td>
<td>Cost Estimate</td>
<td>219</td>
</tr>
<tr>
<td>5.3</td>
<td>Future Effort</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td>REFERENCES</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>APPENDIX I - ANDERSON, BALL, VOSS METHOD</td>
<td>I-1</td>
</tr>
<tr>
<td></td>
<td>APPENDIX II - DERIVATION OF DIFFERENCE EQUATIONS FOR LINEAR FILTERS</td>
<td>II-1</td>
</tr>
<tr>
<td></td>
<td>APPENDIX III - BIT SYNCHRONIZATION ERROR</td>
<td>III-1</td>
</tr>
<tr>
<td></td>
<td>APPENDIX IV - LISTING OF DIGITAL RECEIVER WITHOUT SAMPLE STORAGE BIT</td>
<td>IV-1</td>
</tr>
<tr>
<td></td>
<td>SYNCHRONIZATION</td>
<td></td>
</tr>
<tr>
<td></td>
<td>APPENDIX V - LISTING OF DIGITAL RECEIVER WITH SAMPLE STORAGE BIT</td>
<td>V-1</td>
</tr>
<tr>
<td></td>
<td>SYNCHRONIZATION</td>
<td></td>
</tr>
<tr>
<td></td>
<td>APPENDIX VI - LISTING OF DIGITAL RECEIVER WITH SAMPLE STORAGE BIT</td>
<td>VI-1</td>
</tr>
<tr>
<td></td>
<td>SYNCHRONIZATION (SPLIT-PHASE DATA)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>APPENDIX VII - LISTING OF BASEBAND DIGITAL RECEIVER</td>
<td>VII-1</td>
</tr>
<tr>
<td></td>
<td>APPENDIX VIII - LISTING OF DIGITAL RECEIVER WITH SWEPT FREQUENCY</td>
<td>VIII-1</td>
</tr>
<tr>
<td></td>
<td>ACQUISITION</td>
<td></td>
</tr>
</tbody>
</table>
LIST OF PAGES

Title Page
ii thru xii
1 thru 222
I-1 thru I-3
II-1 thru II-9
III-1 thru III-6
IV-1 thru IV-6
V-1 thru V-11
VI-1 thru VI-11
VII-1 thru VII-5
VIII-1 thru VIII-10
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Linearized Phase-Locked Loop</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>Phase-Locked Loop Block Diagram</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>Differential Equation Implementation</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Finite Difference Model of Phase-Locked Loop</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>Computer Program for Runge-Kutta Method</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>Computer Programs for Euler's and the Z-Transform Method</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>Computer Program for Tustin's and ABV Method</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>Mechanization of the Z-Transform Method</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>Mechanization of Euler's Method</td>
<td>17</td>
</tr>
<tr>
<td>10</td>
<td>Routine for Calculating Sine</td>
<td>18</td>
</tr>
<tr>
<td>11</td>
<td>Accuracy of Sine Wave Generator</td>
<td>20</td>
</tr>
<tr>
<td>12</td>
<td>Comparison of Numerical Methods for a Ramp Input</td>
<td>22</td>
</tr>
<tr>
<td>13</td>
<td>Comparison of Numerical Methods for a Step Input</td>
<td>22</td>
</tr>
<tr>
<td>14</td>
<td>Mechanization of the ABV Method and Tustin's Method</td>
<td>24</td>
</tr>
<tr>
<td>15</td>
<td>Comparing Numerical Methods for Ramp Input as a Function of Computation Time</td>
<td>26</td>
</tr>
<tr>
<td>16</td>
<td>Comparing Numerical Methods for Step Input as a Function of Computation Time</td>
<td>27</td>
</tr>
<tr>
<td>17</td>
<td>Comparison of Numerical Techniques for Linear Filter with Ramp Input</td>
<td>28</td>
</tr>
<tr>
<td>18</td>
<td>Comparison of Numerical Techniques for Linear Filter with Step Sine Wave Input</td>
<td>29</td>
</tr>
<tr>
<td>19</td>
<td>Phase Variance Versus Sample Interval</td>
<td>31</td>
</tr>
<tr>
<td>20</td>
<td>Phase Error Curve (Digital Simulation)</td>
<td>32</td>
</tr>
<tr>
<td>FIGURE NO.</td>
<td>TITLE</td>
<td>PAGE</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>21</td>
<td>Mean Squared Error Versus Offset Frequency for Different Sample Intervals</td>
<td>34</td>
</tr>
<tr>
<td>22</td>
<td>Standard Deviation of Phase Output Versus the Number of Quantization Levels</td>
<td>36</td>
</tr>
<tr>
<td>23</td>
<td>Phase Error Variance Versus the Number of Quantization Levels</td>
<td>37</td>
</tr>
<tr>
<td>24</td>
<td>Scale Factor Determination</td>
<td>39</td>
</tr>
<tr>
<td>25</td>
<td>Phase-Locked Loop with Input Limiter</td>
<td>40</td>
</tr>
<tr>
<td>26</td>
<td>Sawtooth Comparator</td>
<td>42</td>
</tr>
<tr>
<td>27</td>
<td>Phase Error Variance for Limiter</td>
<td>44</td>
</tr>
<tr>
<td>28</td>
<td>Phase Error Variance for Sawtooth Comparator</td>
<td>45</td>
</tr>
<tr>
<td>29</td>
<td>Closed Loop Analog AGC</td>
<td>46</td>
</tr>
<tr>
<td>30</td>
<td>Implementation Diagram for AGC Loop</td>
<td>47</td>
</tr>
<tr>
<td>31</td>
<td>Phase Error Variance Versus Input SNR</td>
<td>49</td>
</tr>
<tr>
<td>32</td>
<td>Swept Frequency Acquisition</td>
<td>51</td>
</tr>
<tr>
<td>33</td>
<td>Determining Relationship Between LPF Bandwidth and Sweep Rate</td>
<td>53</td>
</tr>
<tr>
<td>34</td>
<td>Computer Program for Acquisition</td>
<td>54</td>
</tr>
<tr>
<td>35</td>
<td>Digital Computer Program for FFT Generation</td>
<td>55</td>
</tr>
<tr>
<td>36</td>
<td>Mean Pull-in Time Versus Offset Frequency</td>
<td>58</td>
</tr>
<tr>
<td>37</td>
<td>Variance in Pull-in Time Versus Frequency Offset</td>
<td>59</td>
</tr>
<tr>
<td>38</td>
<td>Probability of Acquisition Versus Sweep Data</td>
<td>60</td>
</tr>
<tr>
<td>39</td>
<td>Optimizing Second Order Input Filter</td>
<td>64</td>
</tr>
<tr>
<td>40</td>
<td>Comparing Input Filters (L Versus R)</td>
<td>65</td>
</tr>
<tr>
<td>41</td>
<td>Comparing Input Filters (K Versus R)</td>
<td>66</td>
</tr>
<tr>
<td>42</td>
<td>Hybrid Configuration</td>
<td>67</td>
</tr>
</tbody>
</table>

viii
<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>43</td>
<td>Tracking Noisy Sine Wave</td>
<td>68</td>
</tr>
<tr>
<td>44</td>
<td>Phase Error Variance without Limiter (Hybrid Simulation)</td>
<td>69</td>
</tr>
<tr>
<td>45</td>
<td>Phase Error Variance with Limiter (Hybrid Simulation)</td>
<td>70</td>
</tr>
<tr>
<td>46</td>
<td>Comparing Digital Simulation Results with Quasi-Linear Approximation</td>
<td>72</td>
</tr>
<tr>
<td>47</td>
<td>Comparing Digital Simulation Results with Fokker-Plank Solution</td>
<td>74</td>
</tr>
<tr>
<td>48</td>
<td>Comparing Measured and Calculated Pull-in Time</td>
<td>76</td>
</tr>
<tr>
<td>49</td>
<td>Comparing Digital Simulation and Theoretical Results for Probability of Acquisition</td>
<td>77</td>
</tr>
<tr>
<td>50</td>
<td>Generation of Noisy PSK Modulated Carrier</td>
<td>80</td>
</tr>
<tr>
<td>51</td>
<td>Direct IF Sampling and A/D Conversion</td>
<td>81</td>
</tr>
<tr>
<td>52</td>
<td>Sample Timing for 1 kHz Operation</td>
<td>83</td>
</tr>
<tr>
<td>53</td>
<td>Sample Timing for 244 Hz Operation</td>
<td>84</td>
</tr>
<tr>
<td>54</td>
<td>DC Bias Remover</td>
<td>85</td>
</tr>
<tr>
<td>55</td>
<td>Digital Signal Generation</td>
<td>90</td>
</tr>
<tr>
<td>56</td>
<td>The Squaring Loop for Phase-Tracking a Binary-Modulated Suppressed-Carrier Input</td>
<td>91</td>
</tr>
<tr>
<td>57</td>
<td>The Costas Loop for Phase-Tracking a Binary PSK Carrier</td>
<td>91</td>
</tr>
<tr>
<td>58</td>
<td>Decision-Oriented Carrier Tracking</td>
<td>92</td>
</tr>
<tr>
<td>59</td>
<td>Bit Error Probability for Costas Loop and Decision Feedback</td>
<td>94</td>
</tr>
<tr>
<td>60</td>
<td>Linearized Costas Loop</td>
<td>96</td>
</tr>
<tr>
<td>61</td>
<td>Phase Error Variance</td>
<td>102</td>
</tr>
<tr>
<td>62</td>
<td>Square Law Bit Synchronizer</td>
<td>105</td>
</tr>
<tr>
<td>FIGURE NO.</td>
<td>TITLE</td>
<td>PAGE</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>63</td>
<td>Delay and Multiply Bit Synchronizer</td>
<td>107</td>
</tr>
<tr>
<td>64</td>
<td>Autocorrelation Function for Output of Delay and Multiply Bit Synchronizer</td>
<td>107</td>
</tr>
<tr>
<td>65</td>
<td>Gate Location</td>
<td>109</td>
</tr>
<tr>
<td>66</td>
<td>Gated Bit Synchronizer</td>
<td>109</td>
</tr>
<tr>
<td>67</td>
<td>Comparison of Bit Synchronizers</td>
<td>111</td>
</tr>
<tr>
<td>68</td>
<td>Bit Synchronizer Acquisition Time</td>
<td>112</td>
</tr>
<tr>
<td>69</td>
<td>Bit Synchronization Loop</td>
<td>114</td>
</tr>
<tr>
<td>70</td>
<td>Loop Error Signal</td>
<td>116</td>
</tr>
<tr>
<td>71</td>
<td>Time Points Defining Early, Late, and Data Integrals</td>
<td>117</td>
</tr>
<tr>
<td>72</td>
<td>Reduced Bit Synchronization Loop</td>
<td>118</td>
</tr>
<tr>
<td>73</td>
<td>Bit Synchronization Feedback Loop</td>
<td>120</td>
</tr>
<tr>
<td>74</td>
<td>Bit Error Rate for Digital Receiver with and without Sample Storage Bit Synchronizer</td>
<td>125</td>
</tr>
<tr>
<td>75</td>
<td>Standard Deviation of Bit Jitter</td>
<td>126</td>
</tr>
<tr>
<td>76</td>
<td>Standard Deviation of Bit Jitter - 5% Timing Error</td>
<td>127</td>
</tr>
<tr>
<td>77</td>
<td>Bit Error Rate - NRZ and Split-Phase</td>
<td>130</td>
</tr>
<tr>
<td>78</td>
<td>Standard Deviation of Bit Jitter - NRZ and Split-Phase</td>
<td>131</td>
</tr>
<tr>
<td>79</td>
<td>Bit Error Rate - NRZ and Split-Phase</td>
<td>132</td>
</tr>
<tr>
<td>80</td>
<td>Standard Deviation of Bit Jitter - NRZ and Split-Phase</td>
<td>133</td>
</tr>
<tr>
<td>81</td>
<td>Standard Deviation of Bit Jitter; AGC and No AGC</td>
<td>135</td>
</tr>
<tr>
<td>82</td>
<td>Digital Receiver with Sample Storage Bit Synchronizer and AGC</td>
<td>136</td>
</tr>
<tr>
<td>FIGURE NO.</td>
<td>TITLE</td>
<td>PAGE</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>83</td>
<td>Costas Loop with Hard Line Sync</td>
<td>138</td>
</tr>
<tr>
<td>84</td>
<td>Flow Chart of Digital PSK Receiver with Data Derived Sync</td>
<td>140</td>
</tr>
<tr>
<td>85</td>
<td>Acquisition Indicator - Simulation</td>
<td>150</td>
</tr>
<tr>
<td>86</td>
<td>Acquisition Indicator - Experimental</td>
<td>151</td>
</tr>
<tr>
<td>87</td>
<td>Loss Caused by Noise Aliasing</td>
<td>159</td>
</tr>
<tr>
<td>88</td>
<td>Loss Caused by Intersymbol Interference</td>
<td>160</td>
</tr>
<tr>
<td>89</td>
<td>Total Sampling Loss</td>
<td>161</td>
</tr>
<tr>
<td>90</td>
<td>Costas Loop Phase Error Variance</td>
<td>166</td>
</tr>
<tr>
<td>91</td>
<td>Probability of Error with a Noisy Phase Reference</td>
<td>168</td>
</tr>
<tr>
<td>92</td>
<td>Standard Deviation of Bit Jitter Versus Signal-to-Noise Ratio</td>
<td>169</td>
</tr>
<tr>
<td>93</td>
<td>Bit Synch Error Versus Signal-to-Noise Ratio (Theoretical)</td>
<td>170</td>
</tr>
<tr>
<td>94</td>
<td>Effect of Sample Rate on Bit Sync Error</td>
<td>172</td>
</tr>
<tr>
<td>95</td>
<td>Bit Error Rate - Sample Storage Bit Sync</td>
<td>173</td>
</tr>
<tr>
<td>96</td>
<td>Bit Error Rate - AGC</td>
<td>175</td>
</tr>
<tr>
<td>97</td>
<td>Bandpass Filter</td>
<td>176</td>
</tr>
<tr>
<td>98</td>
<td>Bit Error Rate - Optimized IF Filter</td>
<td>177</td>
</tr>
<tr>
<td>99</td>
<td>Bit Error Rate Split-Phase Data</td>
<td>178</td>
</tr>
<tr>
<td>100</td>
<td>Reduction of Dimensionality</td>
<td>183</td>
</tr>
<tr>
<td>101</td>
<td>Probability of Bit Error with Bayesian Estimation</td>
<td>191</td>
</tr>
<tr>
<td>102</td>
<td>Typical Learning Curve for Bit Synchronization</td>
<td>192</td>
</tr>
<tr>
<td>103</td>
<td>Phase Estimation Processor</td>
<td>194</td>
</tr>
<tr>
<td>104</td>
<td>Candidate Systems Versus Speed</td>
<td>196</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES (continued)

<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>105</td>
<td>Time Slot Operations</td>
<td>196</td>
</tr>
<tr>
<td>106</td>
<td>Functional Block Diagram</td>
<td>199</td>
</tr>
<tr>
<td>107</td>
<td>Operation Sequencer and Clock</td>
<td>203</td>
</tr>
<tr>
<td>108</td>
<td>Timing Diagram</td>
<td>205</td>
</tr>
<tr>
<td>109</td>
<td>Arithmetic Logic Units A and C</td>
<td>206</td>
</tr>
<tr>
<td>110</td>
<td>Arithmetic Logic Unit B</td>
<td>209</td>
</tr>
<tr>
<td>111</td>
<td>Method A Sine-Cosine ROM</td>
<td>214</td>
</tr>
<tr>
<td>112</td>
<td>Method B Sine-Cosine ROM</td>
<td>216</td>
</tr>
<tr>
<td>113</td>
<td>Input-Output Registers</td>
<td>217</td>
</tr>
<tr>
<td>III1</td>
<td>Bit Synchronization Model</td>
<td>III-1</td>
</tr>
<tr>
<td>III2</td>
<td>Bit Synchronization Integrals</td>
<td>III-4</td>
</tr>
<tr>
<td>III3</td>
<td>Standard Deviation of Bit Jitter Versus</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Signal-to-Noise Ratio</td>
<td></td>
</tr>
</tbody>
</table>
1. **INTRODUCTION**

Potential applications of digital computers in communications systems continue to expand. Many ground stations have ready access to general purpose computers that could be used as data processors. In this study, we have developed the digital signal processing techniques necessary for performing all the functions of an analog communication receiver.

The computer software developed for the MDAC Digital PSK receiver makes it possible to use any general purpose computer (with A/D conversion capability) as a receiver for low data rate telemetry processing. The first step in the receiver processing is sampling and A/D conversion. The sampling can either be performed directly on the IF signal, or on the quadrature components of the IF signal derived from mixing the IF with a frequency oscillator. We have operated our receiver with both sampling techniques without any noticeable difference in performance.

The sampled quadrature components are A/D converted for further processing by a set of digital algorithms which are roughly equivalent to a digital Costas loop. The output of the Costas loop is mixed with the input signal to develop the NRZ video signal. Bit synchronization is performed by a set of digital algorithms which constitute an early-late gate type synchronizer. The bit synchronizer uses four offset integrators to perform phase detection along with a digital phase locked loop for tracking. A digital integrator and threshold is then used for bit detection.

The logic speed of general purpose computers makes this software useful for data rates of 0-100 bits/second. Very low bit rates are difficult to demodulate with analog processors due to the need for very long time constant filters, precise component values, and very stable timing references. Thus,
the digital software discussed in this report provide a low cost, efficient, and versatile solution to the low data rate receiver problem. This software can also easily be modified to incorporate any data format desired. Measurements of receiver performance (i.e., probability of error/bit) indicate that the Digital PSK Receiver will operate as close to the theoretical optimum as the logic speed and tracking bandwidths will permit.

The Digital Receiver Study and Implementation conducted under contract NAS5-11424 for NASA GSFC, included the optimization of the MDAC digital PSK receiver software (developed under contract NAS5-21021), an expanded study of digital bit synchronization algorithms, and a preliminary study of digital processing techniques which have no real analog equivalent. This report also contains the principal results from the previous contract NAS5-21021, the study of Digital Phase Lock Techniques. The results of both contracts were combined since they are highly interrelated and are both incomplete without the other. All system simulations for both contracts were performed on the MDAC CDC 6400 and CDC 6600 computers, and all receiver software was optimized for the GSFC CDC 3200 computer.
2. DEVELOPMENT OF DIGITIZED PHASE-LOCKED LOOP

This effort consisted of designing, implementing, and evaluating a digitized phase-locked loop capable of tracking a carrier imbedded in wide band noise. As part of the design effort, we compared the various numerical techniques available for implementing a digital filter, synthesized and analyzed the loop equations, and evaluated the performance by determining curves of phase error variance as a function of the signal-to-noise ratio, sampling rate, and quantization interval size. We also investigated digital techniques for implementing AGC and acquisition circuits.

2.1 Synthesis of Equations

We selected a phase-locked loop filter by considering its effect on transient response (damping), bandwidth, and steady state tracking-offset for a ramp input. Utilizing this filter, we analyzed the linear loop in order to determine the filter and gain constants as a function of the damping factor (ζ), and the undamped natural frequency (ω₀). Finally, we analyzed the non-linear loop and developed numerical equivalents for it using both the differential and difference equation approach.

2.1.1 Choice of Phase-Locked Loop Filter

The filter in a second order phase-locked loop can have any of the forms given in Equations (1-3).

\[ F_1(s) = \frac{b}{s+b} \]  \hspace{1cm} (1)

\[ F_2(s) = \left( \frac{b}{a} \right) \frac{(s+a)}{(s+b)} \]  \hspace{1cm} (2)

\[ F_3(s) = \frac{s+a}{s} \]  \hspace{1cm} (3)
We determined the effect of the three phase-locked loop filters on the transient response (damping), bandwidth, and steady state tracking offset for a ramp input to the associated phase-locked loop. The filter represented by Equation (1) was not chosen because the bandwidth and phase offset both depend on the loop gain and thus cannot be chosen independently. If the filter given in Equation (2) is utilized, the filter constant (a) and the loop gain are chosen to give the proper bandwidth and damping, and the filter constant (b) is chosen to give an acceptable phase offset. Since the phase offset will normally be made small, the constant (b) for \( F_2(s) \) will be near zero, and \( F_3(s) \) and \( F_2(s) \) are essentially the same. Therefore, the choice must be made in terms of implementation complexity. Since the digital implementation of \( F_3(s) \) requires one less multiplication, it was selected as the phase-locked loop filter. \( F_2(s) \) would be our choice for an analog implementation because of the difficulty associated with implementing the open loop integrator associated with \( F_3(s) \). This decision is also supported by the fact that for an input consisting of a ramp of phase the third filter form minimizes the mean squared error at the loop output.

2.1.2 Analysis of Linear Loop

A block diagram of the linearized phase-locked loop is given in Figure 1 where the input signal is assumed to be of the form \( A \sin(\omega_0 t + \theta_{in}) \).
The loop transfer function can then be determined as shown in Equations (4) and (5).

\[
\frac{\theta_{\text{out}}}{\theta_{\text{in}}} = \frac{AK(s+a)}{s^2 + AK(s+a)} \quad (4)
\]

\[
\frac{\theta_{\text{out}}}{\theta_{\text{in}}} = \frac{AK(s+a)}{s^2 + AKs + aAK} \quad (5)
\]

This transfer function can then be put in the standard form given in Equation (6).

\[
\frac{\theta_{\text{out}}}{\theta_{\text{in}}} = \frac{2\xi\omega_0 s + \omega_0^2}{s^2 + 2\xi\omega_0 s + \omega_0^2} \quad (6)
\]

\[AK = 2\xi\omega_0 \quad (7)\]

\[aAK = \omega_0^2 \quad (8)\]

If the input amplitude, the damping factor (\(\xi\)), and the undamped natural frequency (\(\omega_0\)) are known, the phase-locked loop gain and filter constant can be determined from Equations (7) and (8) and are given in Equations (9) and (10).

\[K = \frac{2\xi\omega_0}{A} \quad (9)\]

\[a = \omega_0/2\xi \quad (10)\]

The response of the above loop to a ramp input of slope \(\omega\) is given in Equation (11).

\[
\theta_{\text{out}}(t) = \omega t - \frac{2\omega^2}{\omega_0 \sqrt{1-\xi^2}} e^{-\delta\omega_0 t} \sin \omega_0 \sqrt{1-\xi^2} t \quad (11)
\]

The above result shows that the effective time constant of the loop is \(\frac{1}{\delta\omega_0}\) seconds. The noise and 3 db bandwidth of the above loop are given in Equations (12) and (13).
Noise Bandwidth = \( B_N = \omega_0 \left( \frac{2\zeta + \frac{1}{2\zeta}}{4} \right) \) Hz

3 dB Bandwidth = \( \frac{\omega_0}{2\pi} \sqrt{(2\zeta^2 + 1) + \sqrt{(2\zeta^2 + 1)^2 + 1}} \) Hz

2.1.3 Differential Equation Approach

With the loop parameters established in terms of standardized functions, we proceeded to develop the numerical equivalents for the differential equations which describe the loop operation. From the block diagram of the phase-locked loop presented in Figure 2, the loop equations can be derived as shown in Equations (14-17).

\[
e = 2(X \sin \omega_0 t + Y \cos \omega_0 t) \cos (\omega_0 t + \theta)
\]

\[
e = -X \sin \theta + Y \cos \theta + \text{(second harmonic terms)}
\]

\[
\frac{d^2 \theta}{dt^2} = K \left( \frac{d}{dt} + a \right) e
\]
\[
\frac{d^2 \theta}{dt^2} = aK \left[ Y \cos \theta - x \sin \theta \right] + K \frac{d}{dt} \left[ Y \cos \theta - X \sin \theta \right] \tag{17}
\]

To avoid taking the derivative of the quadrature components, the above equation can then be converted to the form shown in Equations (18) and (19).

\[
\frac{d\theta}{dt} = aK (Y \cos \theta - X \sin \theta) \tag{18}
\]

\[
\frac{d\theta}{dt} = c + K (Y \cos \theta - X \sin \theta) \tag{19}
\]

One discrete time technique that was used to solve the above equations is the Runge-Kutta Method. Using results obtained from Scarborough, the equations for this technique are shown in Figure 3. A second technique investigated was Euler's Method. The equations for this method are also given in Figure 3.

A comparison of these and other techniques is presented in Section 2.3.
## Differential Equation Implementation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Input sine wave amplitude</td>
</tr>
<tr>
<td>h</td>
<td>Time increment</td>
</tr>
<tr>
<td>(\theta_n)</td>
<td>Output phase at time (t_n)</td>
</tr>
<tr>
<td>(C_n)</td>
<td>Dummy variable (C) at time (t_n)</td>
</tr>
<tr>
<td>(X_n', Y_n)</td>
<td>Quadrature components of input signal at time (t_n)</td>
</tr>
<tr>
<td>(K)</td>
<td>(2\zeta\cos\alpha/A)</td>
</tr>
<tr>
<td>(a)</td>
<td>(\omega_0/2\zeta)</td>
</tr>
</tbody>
</table>

### Runge-Kutta Method

\[
F(\theta, X, Y) = Kh (Y \cos \theta - X \sin \theta)
\]

\[
G_1 = F(\theta_{n-2}, X_{n-2}, Y_{n-2})
\]

\[
A_1 = h C_{n-2} + G_1
\]

\[
B_1 = a G_1
\]

\[
G_2 = F(\theta_{n-2} + A_1/2, X_{n-1}, Y_{n-1})
\]

\[
A_2 = h [C_{n-2} + B_1/2] + G_2
\]

\[
B_2 = a G_2
\]

\[
G_3 = F(\theta_{n-2} + A_2/2, X_{n-1}, Y_{n-1})
\]

\[
A_3 = h [C_{n-2} + B_2/2] + G_3
\]

\[
B_3 = a G_3
\]

\[
G_4 = F(\theta_{n-2} + A_3, X_n, Y_n)
\]

\[
A_4 = h [C_{n-2} + B_3] + G_4
\]

\[
B_4 = a G_4
\]

\[
C_n = C_{n-2} + (B_1 + 2B_2 + 2B_3 + B_4)/6
\]

\[
\theta_n = \theta_{n-2} + (A_1 + 2A_2 + 2A_3 + A_4)/6
\]

### Euler's Method

\[
G = Y_{n-1} \cos \theta_{n-1} - X_{n-1} \sin \theta_{n-1}
\]

\[
\theta_n = \theta_{n-1} + h C_{n-1} + KhG
\]

\[
C_n = C_{n-1} + a KhG
\]
2.1.4 Difference Equation Formulation

The difference equation approach is basically a method of determining the present value of the phase-locked loop output in terms of past output values and past and present input values. In order to formulate the difference equations for the phase-locked loop, the block diagram of Figure 4 is utilized.

FINITE DIFFERENCE MODEL OF PHASE-LOCKED LOOP

\[ X_n \sin \omega_0 t_n + Y_n \cos \omega_0 t_n \]
\[ 2 \cos (\omega_0 t_n + \theta_n) \]
\[ \theta_n \]

\[ e_n = 2(\sin \omega_0 t_n + Y \cos \omega_0 t_n) \cos (\omega_0 t_n + \theta_n) \]  
\[ e_n = -X_n \sin \theta_n + Y_n \cos \theta_n + \text{(second harmonic terms)} \]

The difference equations for the above model can be written as shown in Equations (20-23).

\[ e_n = 2(X_n \sin \omega_0 t_n + Y_n \cos \omega_0 t_n) \cos (\omega_0 t_n + \theta_n) \]  
\[ e_n = -X_n \sin \theta_n + Y_n \cos \theta_n + \text{(second harmonic terms)} \]

\[ \theta(z) = \frac{A_2 z^2 + A_1 z + A_0}{B_2 z^2 + B_1 z + B_0} \]  
\[ e(z) = \frac{1}{B_2} \left( A_2 e_n + A_1 e_{n-1} + A_0 e_{n-2} - B_1 \theta_{n-1} - B_0 \theta_{n-2} \right) \]

The first technique we investigated for implementing difference equations was the Z-Transform method. In order to use this technique a hold circuit must be placed in front of the phase-locked loop filter. \( F(z) \) can then be determined as shown

\[ F(z) = (1-z^{-1}) \left( \frac{Ks + aK}{s^3} \right) \]
The difference equation for the loop can be written as shown in Equations (26-28).

\[ e_{n-1} = Y_{n-1} \cos \Theta_{n-1} - X_{n-1} \sin \Theta_{n-1} \]  
\[ e_{n-2} = Y_{n-2} \cos \Theta_{n-2} - X_{n-2} \sin \Theta_{n-2} \]  
\[ \Theta_n = 2\Theta_{n-1} - \Theta_{n-2} + \frac{1}{2} (2K_h + aK_h^2) e_{n-1} + \frac{1}{2} (aK_h^2 - 2K_h) e_{n-2} \]  

The second technique investigated was Tustin's Method. This technique corresponds to trapezoidal rule integration. In order to obtain the difference equation from the transfer function, the integration operators are replaced by

\[ \frac{1}{s} \rightarrow \frac{h(1 + z^{-1})}{2(1 - z^{-1})} \]  
\[ \frac{1}{s^2} \rightarrow \frac{h^2(1 + z^{-1})^2}{4(1 - z^{-1})^2} \]

F(z) can then be obtained

\[ F(s) = \frac{\Theta(s)}{e(s)} = \frac{Ks + aK}{s^2} = K \left( \frac{1}{s} \right) + aK \left( \frac{1}{s^2} \right) \]  
\[ \frac{1}{s^2} = \frac{h^2(1 + z^{-1})^2}{4(1 - z^{-1})^2}, \quad \frac{1}{s} = \frac{h(1 + z^{-1})}{2(1 - z^{-1})} \]

\[ F(z) = \frac{\Theta(z)}{e(z)} = \frac{2K_h(z - 1) + aK_h^2(z^2 + 2z + 1)}{4(z^2 - 2z + 1)} \]

Since the phase output \( \Theta_n \) depends on \( e_n \), a unit delay must be added in the feedback path. The results are given in Equations (34-37).
The final technique considered was the Anderson, Ball, Voss Method. This technique consists of approximating the input signal by a polynomial in $t$ and then computing the response at the output of the filter. The input signal was approximated by a second order polynomial, and the resulting difference equation is given in Equations (38-41). The unit delay is again added in the feedback path because of the problem mentioned with respect to Tustin's Method. Because of the complexity associated with this technique, the derivation of this result appears in Appendix I.

$$e_n = Y_n \cos \theta_{n-1} - X_n \sin \theta_{n-1} \quad (38)$$

$$e_{n-1} = Y_{n-1} \cos \theta_{n-2} - X_{n-1} \sin \theta_{n-2} \quad (39)$$

$$e_{n-2} = Y_{n-2} \cos \theta_{n-3} - X_{n-2} \sin \theta_{n-3} \quad (40)$$

$$\theta_n = 2\theta_{n-1} - \theta_{n-2} + \frac{1}{4} \left(2Kh + aKh^2\right) e_n + \frac{1}{2} aKh^2 e_{n-1} + \frac{1}{4} \left(aKh^2 - 2Kh\right) e_{n-2} \quad (41)$$

Equations (38-41) show that the Anderson, Ball, Voss technique gives results that are almost identical with those obtained using Tustin's Method. Therefore, for this particular filter, the higher order approximation used with the Anderson, Ball, Voss technique does not increase the accuracy of the result. We also investigated the Madwed-Truxal and the Boxer-Thaler Techniques. These
methods are similar to Tustin's method except that different approximations are used for the higher order integration operators. Our results showed that these techniques give only an insignificant increase in accuracy and are more unstable than Tustin's method. Therefore, it was concluded that these techniques did not merit further consideration.

2.2 Development of Computer Programs

We developed digital computer programs for the five techniques considered in Section 2.1. These programs were written in the Fortran IV language which is compatible with the CDC 3200 computer. The Runge-Kutta Method is shown in Figure 5; Euler's Method and the Z-Transform technique is shown in Figure 6; and Tustin's Method and the Anderson, Ball, Voll (ABV) Method is illustrated in Figure 7. In these figures ADC (01) and ADC (02) represent the digitized values of the sampled quadrature components which have been sent through the analog to digital converters.

Mechanization block diagrams for the Z-Transform technique and for Euler's method are shown in Figures 8 and 9 respectively. As is shown in the following section, these two approaches give near optimum results for the non-linear phase-locked loop. The blocks containing $Z^{-1}$ represent a storage register which delays the signal by one sample period.

The most complex and time consuming portion of these implementations is the sine and cosine calculation. One possible method of simplifying this calculation is to compute new values of the sine and cosine from the previous values by using the small angle approximation as shown in Equations (42) and (43).
### COMPUTER PROGRAM FOR RUNGE–KUTTA METHOD

| ADC(01) = DIGITIZED VALUE OF X QUADRATURE COMPONENT FROM ANALOG CIRCUITRY |
| ADC(02) = DIGITIZED VALUE OF Y QUADRATURE COMPONENT FROM ANALOG CIRCUITRY |
| H = TIME BETWEEN SAMPLES |
| AK = KH |
| AA = a |
| C5 = H/2 |
| T = \( \theta_n \) |
| TM1 = \( \theta_{n-1} \) |
| TM2 = \( \theta_{n-2} \) |
| C = Cn |
| CM1 = C_{n-1} |
| CM2 = C_{n-2} |
| XM = x_n |
| YM = y_n |
| SM1 = x_{n-1} |
| YMI = y_{n-1} |
| XM2 = x_{n-2} |
| YM2 = y_{n-2} |

| X4 = ADC(01) |
| Y4 = ADC(02) |
| C3 = CM2 * H |
| C2 = AK * (XM2 * SIN(TM2) - YM2 * COS(TM2)) |
| A1 = C3 + C2 |
| B1 = AA * C2 |
| C4 = TM2 + .5 * A1 |
| C2 = AK * (XM1 * SIN(C4) - YMI * COS(C4)) |
| A2 = C3 + C6 * B1 + C2 |
| B2 = AA * C2 |
| XM = XM2 |
| YM = YM2 |
| CM2 = CM1 |
| CMI = C |

**FIGURE 5**
### Euler's Method

<table>
<thead>
<tr>
<th>Variable</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>TIME BETWEEN SAMPLES</td>
</tr>
<tr>
<td>$AK$</td>
<td>$KH$</td>
</tr>
<tr>
<td>$AA$</td>
<td>$a$</td>
</tr>
<tr>
<td>$T_o$</td>
<td>$T_n$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$ADC(01)$</td>
</tr>
<tr>
<td>$Y$</td>
<td>$ADC(02)$</td>
</tr>
<tr>
<td>$\text{TEMP}$</td>
<td>$AK \cdot (Y \cdot \cos(T) - X \cdot \sin(T))$</td>
</tr>
<tr>
<td>$T$</td>
<td>$T + C \cdot H + \text{TEMP}$</td>
</tr>
<tr>
<td>$C$</td>
<td>$C + AA \cdot \text{TEMP}$</td>
</tr>
</tbody>
</table>

### Z-Transform Method

<table>
<thead>
<tr>
<th>Variable</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>TIME BETWEEN SAMPLES</td>
</tr>
<tr>
<td>$A1$</td>
<td>$KH + \frac{1}{2}aKH^2$</td>
</tr>
<tr>
<td>$A2$</td>
<td>$\frac{1}{2}aKH^2 - KH$</td>
</tr>
<tr>
<td>$T$</td>
<td>$T_n$</td>
</tr>
<tr>
<td>$TM1$</td>
<td>$T_{n-1}$</td>
</tr>
<tr>
<td>$TM2$</td>
<td>$T_{n-2}$</td>
</tr>
<tr>
<td>$EM1$</td>
<td>$E_{n-1}$</td>
</tr>
<tr>
<td>$EM2$</td>
<td>$E_{n-2}$</td>
</tr>
<tr>
<td>$XM1$</td>
<td>$X_{n-1}$</td>
</tr>
<tr>
<td>$YM1$</td>
<td>$Y_{n-1}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$XM1$</td>
<td>$ADC(01)$</td>
</tr>
<tr>
<td>$YM1$</td>
<td>$ADC(02)$</td>
</tr>
<tr>
<td>$EM1$</td>
<td>$YM1 \cdot \cos(TM1) - XM1 \cdot \sin(TM1)$</td>
</tr>
<tr>
<td>$T$</td>
<td>$2 \cdot TM1 - TM2 + A1 \cdot EM1 + A2 \cdot EM2$</td>
</tr>
<tr>
<td>$TM2$</td>
<td>$TM1$</td>
</tr>
<tr>
<td>$TM1$</td>
<td>$T$</td>
</tr>
<tr>
<td>$EM2$</td>
<td>$EM1$</td>
</tr>
</tbody>
</table>
## COMPUTER PROGRAM FOR TUSTIN'S AND ABV METHOD

### TUSTIN'S METHOD

- \( T = \theta_n \)
- \( TM1 = \theta_{n-1} \)
- \( TM2 = \theta_{n-2} \)
- \( EM1 = e_{n-1} \)
- \( EM2 = e_{n-2} \)
- \( XM = x_{n-1} \)
- \( YM = y_{n-1} \)

\[
\begin{align*}
H &= \text{TIME BETWEEN SAMPLES} \\
C1 &= KH/2 + aKH^2/2 \\
C2 &= aKH^2/2 \\
C3 &= aKH^2/4 - KH/2 \\
XM &= \text{ADC(01)} \\
YM &= \text{ADC(02)} \\
EM &= YM \times \cos(TM1) - XM \times \sin(TM1) \\
T &= 2 \times TM1 - TM2 + C1 \times EM + C2 \times EM1 + C3 \times EM2 \\
EM2 &= EM1 \\
EM1 &= EM \\
TM2 &= TM1 \\
TM1 &= T
\end{align*}
\]

### ANDERSON, BALL, VOSS METHOD (ABV)

- \( H = \text{TIME BETWEEN SAMPLES} \)
- \( C1 = KH/2 + aKH^2 \)
- \( C2 = 5aKH^2/6 \)
- \( C3 = aKH^2/12 - KH/2 \)

\[
\begin{align*}
XM &= \text{ADC(01)} \\
YM &= \text{ADC(01)} \\
EM &= YM \times \cos(TM1) - XM \times \sin(TM1) \\
T &= 2 \times TM1 - TM2 + C1 \times EM + C2 \times EM1 + C3 \times EM2 \\
EM2 &= EM1 \\
EM1 &= EM \\
TM2 &= TM1 \\
TM1 &= T
\end{align*}
\]
MECHANIZATION OF THE Z-TRANSFORM METHOD

\[ x_n \xrightarrow{} \Sigma \xrightarrow{z^{-1}} e_n \xrightarrow{} e_{n-1} \xrightarrow{} z^{-1} \frac{1}{2} \omega_n h (4\delta + \omega_n h) \xrightarrow{} \frac{1}{2} \omega_n h (\omega_n h - 4\delta) \]

\[ y_n \xrightarrow{} \Sigma \xrightarrow{} \Sigma \xrightarrow{} \Sigma \xrightarrow{z^{-1}} \theta_{n-1} \xrightarrow{2} z^{-1} \]

FIGURE 8
MECHANIZATION OF EULER'S METHOD

\[
\cos (\theta + \Delta \theta) = \cos \theta \cos \Delta \theta - \sin \Delta \theta \sin \theta \quad (42)
\]
\[
\sin (\theta + \Delta \theta) = \sin \Delta \theta \cos \theta + \cos \Delta \theta \sin \theta \quad (43)
\]

If \( \Delta \theta \) is small, this result can be simplified as shown in Equations (44) and (45).

\[
\cos (\theta + \Delta \theta) = \cos \theta - \Delta \theta \sin \theta \quad (44)
\]
\[
\sin (\theta + \Delta \theta) = \sin \theta + \Delta \theta \cos \theta \quad (45)
\]

Starting with an initial value for \( \sin \theta \) and \( \cos \theta \), the above formulas can be used to recursively evaluate the sine and cosine if \( \Delta \theta \) is small. However, Larimore \(^2\) shows that this technique will become unstable unless a correction
factor is applied to the above calculation. He suggests that this correction factor make the sum of the squares of the sine and cosine equal to one as shown in Equations (46-51).

\[ K = \text{correction factor} \]

\[ K^2 \left[ \cos^2 (\theta + \Delta \theta) + \sin^2 (\theta + \Delta \theta) \right] = 1 \]  \hspace{1cm} (46)

\[ K = \frac{1}{\sqrt{\cos^2 (\theta + \Delta \theta) + \sin^2 (\theta + \Delta \theta)}} \]  \hspace{1cm} (47)

let \( E = \cos^2 (\theta + \Delta \theta) + \sin^2 (\theta + \Delta \theta) - 1 \)  \hspace{1cm} (48)

\[ \therefore K = \frac{1}{\sqrt{1 + E}} = (1 + E)^{-\frac{1}{2}} \]  \hspace{1cm} (49)

if \( E \approx 0 \)

\[ K \approx 1 - \frac{1}{2} E \]  \hspace{1cm} (50)

A Fortran program for continuously making the above calculation is given in Figure 10.

**ROUTINE FOR CALCULATING SINE**

```
CMP = CM - DTHE * SMI
SMP = SMI + DTHE * CMP
DN = 1.5 - 0.5 * (SMP ** 2 + CMP ** 2)
CM = DN * CMP
SM = DN * SMP
CMI = CM
SMI = SM
```

**FIGURE 10**
Utilizing the CDC 6400 computer, a timing calculation was made for both the above routine and the standard subroutine used in the 6400 for computing the sine and cosine. The results showed that the simple technique required 49 μsec of central processor time per iteration, while the more complex method required 251 μsec per iteration. Although the simple technique is 5 times faster, its accuracy degrades as a function of both the number of calculations and the increment size, as shown in Figure 11. If we sample at a rate ten times the maximum offset frequency, the average output phase change would be 36°. Therefore, from the results given in Figure 11, it would be necessary to update the sine and cosine generator after only a small number of iterations, thus making the usefulness of this technique questionable. One possible method of updating the calculation would be to store several values of the sine and cosine uniformly spaced between zero and ninety degrees, and then selecting the nearest angle after a fixed number of calculations. In order to implement this approach logic statements must be added to the program, which would result in an increased computation time. A transient will also occur in the phase-locked loop each time an update is made. A better solution to this problem might be to write a more efficient routine for generating the sine and cosine or to use a table look-up technique. A trade-off between accuracy and speed could then be made to determine the optimum approach. Designing an optimum sine and cosine generator for use in this phase-locked loop program represents a prime area for further study.
ACCURACY OF SINE WAVE GENERATOR

\[ \sqrt{\text{MEAN SQUARED ERROR}} \]

\[ N = \text{NUMBER OF CALCULATIONS} \]

- \( N = 500 \)
- \( N = 200 \)
- \( N = 100 \)
- \( N = 50 \)
- \( N = 20 \)
- \( N = 10 \)

\[ \text{ANGLE INCREMENT - DEGREES} \]

FIGURE 11
2.3 Comparison of Techniques

In this section the five selected numerical techniques are compared on the basis of accuracy and speed for two linear second order filters and a phase-locked loop with various inputs. The second order filter and phase-locked loop were chosen because they represent a typical linear and non-linear subsystem component. The analog linear filter output can easily be calculated and thus the error of the discrete filter measured.

2.3.1 Non-Linear Phase-Locked Loop

As a first step in comparing the five numerical methods described in Sections 2.1 and 2.2, we determined the central processor time required for one iteration of each of the candidate techniques using the standard machine subroutine for sin and cos. These times, which apply to our CDC 6400 computer, are given below.

- Euler 240.3 μsec
- Tustin 273.7 μsec
- Anderson, Ball, Voss 273.7 μsec
- Z-Transform 311.9 μsec
- Runge-Kutta 1334.0 μsec

We also determined the mean-squared error between the output of a continuous loop and the digitized implementation for both ramp and step inputs. The ramp input signal was started at a sample point, while the step input was begun at a point halfway between the sample points. The results of this comparison are given in Figures 12 and 13. These curves indicate that Tustin's Method and the Anderson, Ball, Voss Method, two techniques which normally give high accuracy, show results considerably below the other candidate approaches. The reason is that a unit delay must be added in the feedback path because of the non-linearity associated with the phase-locked loop. In Section 2.3.2, the above techniques are compared for two linear filters, and the increased accuracy of Tustin's Method and the ABV Method is evident.
COMPARISON OF NUMERICAL METHODS FOR A RAMP INPUT

\[ S_E = 10 \log \left( \frac{1}{\text{ERROR}} \right) \]

\[ \zeta = 0.707 \]

\[ \omega_0 = 1.414 \text{ RAD/SEC} \]

FIGURE 12

COMPARISON OF NUMERICAL METHODS FOR A STEP INPUT

\[ S_E = 10 \log \left( \frac{1}{\text{ERROR}} \right) \]

\[ \zeta = 0.707 \]

\[ \omega_0 = 1.414 \text{ RAD/SEC} \]

FIGURE 13
The complexity of the candidate techniques was assessed by drawing block diagrams, using the unit delay element, showing how the different methods can be implemented. Diagrams for Euler's Method and the Z-Transform Method were given previously in Figures 8 and 9. A block diagram showing the implementation of Tustin's Method and the ABV Method is given in Figure 14. One diagram suffices for these two techniques since they differ only by the value of the constants.

The numerical techniques described above were then compared using the preceding information on accuracy, complexity, and speed as the criteria of goodness. The Runge-Kutta Method was immediately discarded because of its complexity and slowness. Tustin's Method and the ABV Method were discarded because of low accuracy. This leaves the Z-Transform Method and Euler's Method, two techniques which give very similar performance as is evidenced by the preceding results. A more quantitative indication of this fact can be obtained by determining the central processor times (per .2 second interval) for a constant value of error. The Z-Transform technique with a sample time of 1/5 the loop time constant was chosen as a reference. This choice was made because 1/5 the loop time constant is a near optimum sample time for the phase-locked loop as will be shown in the next section, and also because this point is on the linear portion of Figure 12 and 13. The results are given below for both the step and ramp input. Note that for a fixed error the Z-Transform requires a slightly greater central processor usage than Euler's Method for a step input and slightly less usage for a ramp input.

**Ramp Input**
- Z-Transform = 311.9 μsec
- Euler = 331.5 μsec

**Step Input**
- Z-Transform = 311.9 μsec
- Euler = 292.3 μsec
MECHANIZATION OF THE ABV METHOD
AND TUSTIN'S METHOD

\[
\theta_n = 26_n - \theta_{n-2} + A_1e_n + A_2e_{n-1} + A_3e_{n-2}
\]

FIGURE 14
A more complete comparison of the numerical techniques is given in Figures 15 and 16 where the signal to error ratio is plotted versus the ratio of sample time to computation time. The above results indicate the near equality between the techniques in terms of speed and accuracy. Since Euler's Method is slightly easier to implement than the Z-Transform Method (the former technique requires three unit delay elements whereas the latter necessitates four), Euler's technique was selected and is used to determine the simulation results given in subsequent sections.

2.3.2 Linear Filters

We also applied the five numerical techniques described above to the two second order filters given in Equations (52) and (53).

\[ H_2(s) = \frac{2\zeta\omega_0 s + \omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \] (52)

\[ H_1(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \] (53)

The difference equations for each of the numerical techniques is derived in Appendix II for \( H_1(s) \) and \( H_2(s) \). Utilizing these formulas, the central processor time for the CDC 6400 computer was determined for each of the methods and is given below.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Processor Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler</td>
<td>35.0 μsec</td>
</tr>
<tr>
<td>Z-Transform</td>
<td>50.9 μsec</td>
</tr>
<tr>
<td>Tustin</td>
<td>62.9 μsec</td>
</tr>
<tr>
<td>ABV</td>
<td>62.9 μsec</td>
</tr>
<tr>
<td>Runge Kutta</td>
<td>214.5 μsec</td>
</tr>
<tr>
<td></td>
<td>274.1 μsec</td>
</tr>
</tbody>
</table>
COMPARING NUMERICAL METHODS FOR RAMP INPUT AS A FUNCTION OF COMPUTATION TIME

INPUT = t(t)
SE = 10 LOG (1/ERROR^2)
ζ = 0.707
ωn = 1.414 RAD/SEC

SAMPLE TIME/COMPUTATION TIME

FIGURE 15
COMPARING NUMERICAL METHODS FOR STEP INPUT AS A FUNCTION OF COMPUTATION TIME

INPUT = u(t)

\[ S_E = 10 \log \left( \frac{1}{\text{ERROR}^2} \right) \]

\[ \zeta = 0.707 \]

\[ \omega_0 = 1.414 \text{ RAD/s} \]

FIGURE 16
We also determined the mean-squared error between the output of the continuous filter and its digitized equivalent for ramp and step sine wave inputs, where the step occurred at a sample point. Since the filter is linear, the output of the continuous filter with the above inputs was calculated. Graphs showing the signal to error ratio versus sampling rate for the two filters are shown in Figures 17 and 18. Thus for a linear system Anderson, Ball, Voss Method is significantly better than Euler's Method due to greater accuracy. However, the non-linearity in the phase lock loop makes Anderson, Ball, Voss inappropriate for our application.

### COMPARISON OF NUMERICAL TECHNIQUES FOR LINEAR FILTER WITH RAMP INPUT

- **INPUT** = \(4.8t(t)\)
- \(S_E = 10 \log (1/\text{ERROR}^2)\)
- \(\zeta = 0.707\)
- \(\omega_0 = 1.414 \text{ RAD/s}\)

**Figure 17**
COMPARISON OF NUMERICAL TECHNIQUES FOR LINEAR FILTER WITH STEP SINE WAVE INPUT

$H(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2}$

INPUT = $\sin(2\pi t)$

$\omega^2 =$ MEAN SQUARED ERROR

$S_E = 10 \log (1/4\omega^2)$

$\zeta = .707$

$\omega_0 = 2\pi$

ANDERSON, BALL, VOSS

RUNGE - KUTTA

TUSTIN

Z - TRANSFORM

EULER

 SIGNAL TO ERROR RATIO - DB

120

100

80

60

40

20

0

-20

-40

10^{-3}

$10^{-2}$

$10^{-1}$

$10^0$

SAMPLE TIME - SEC

FIGURE 18
2.4 Determination of System Constraints

The effect of sampling rate, quantization noise, and dynamic range on digital phase-locked loop performance is determined in this section.

2.4.1 Sampling Rate

The necessary sampling rate depends on the numerical technique being used, the bandwidth of the phase-locked loop, and the offset frequency at the input to the loop. Since Euler's Method has already been selected as the numerical method, only the latter two effects will be considered here. The effect of sampling rate was determined by measuring phase variance versus sample interval for a fixed input noise spectral density using an all digital simulation. The input noise was generated using a Gaussian random number generator. It was assumed for all digital simulations in Section 2 of this study that the quadrature components were prefiltered with an integrate and dump filters. The resulting curve, Figure 19, shows that for the phase-locked loop being considered ($\zeta = 0.707$ and $\omega_o = 1.414$ rad/sec) the sample interval must be less than 70% of the loop time constant to keep the loop from going unstable. We also determined a graph of phase variance versus $\frac{(N_B n)}{A^2}$, the loop signal-to-noise ratio, as a function of the sampling rate using an all digital simulation. This graph, which is shown in Figure 20, indicates that a serious degradation occurs for sample intervals between 20% and 40% of the loop time constant. Therefore, a sampling interval of 20% of the loop time constant represents a nearly optimum selection for the case of no offset frequency since this value is well below the point of instability and also allows the digital phase-locked loop to operate at a point where its transient performance approaches that of the continuous analog loop. We next determined the effect of an offset frequency on the sampling rate requirements. This was accomplished by computing the mean squared value of the difference between the continuous loop output and the
FIGURE 19

\[
\xi = 0.707 \\
\alpha_0 = 1.314 \text{ RAD/s} \\
N_o B W = -1 \\
A^2 = 1
\]
PHASE ERROR CURVE (DIGITAL SIMULATION)

$H = \text{SAMPLE INTERVAL}$

- $H = 0.4$
- $H = 0.2$
- $H = 0.1$
- $H = 0.05$

SOLUTION FOR LINEARIZED ANALOG LOOP

$\text{PHASE VARIANCE - RAD}^2$

$N_o B_n/A^2$

FIGURE 20
digitized loop for different frequency offsets. A graph showing the error versus the offset frequency for two different sampling rates is given in Figure 21. This curve shows that decreasing the sampling interval from 20% to 10% of the loop time constant does not increase the accuracy significantly and also does not allow operation at a much higher offset frequency. Therefore, if the offset frequency is within the bandwidth of the loop, a sampling interval of 20% of the loop time constant gives adequate results. For frequency offsets much greater than the loop bandwidth, the sampling rate is dependent on both the input filter and the frequency of the offset. This dependency will be given more attention in the section on input filters.

2.4.2 Quantization

In order to determine the effect of quantization on the operation of the phase-locked loop, we analytically determined the phase error variance as a function of the number of quantization levels. If the quantization error is assumed to be uniform, the noise variance caused by this error is given by the following formula.

$$\sigma_Q^2 = \frac{a^2}{12}$$  \hspace{1cm} (54)

$$a = \frac{V_p}{N}$$  \hspace{1cm} (55)

$$V_p = \text{peak voltage}$$

$$N = \text{number of positive quantization levels}$$

If it is assumed that adjacent samples of the quantization noise are independent, the phase variance at the output of the linearized phase-locked loop can be determined as shown below. It is also assumed that the noise samples are held for one sample interval.

$$\sigma_{\phi}^2 = \frac{N_B}{A^2}$$  \hspace{1cm} (56)

$$N_B = \text{Spectral density of quantization noise}$$

$$N_B = 2 \sigma_Q^2 h$$
MEAN SQUARED ERROR VERSUS OFFSET FREQUENCY FOR DIFFERENT SAMPLE INTERVALS

$H = \text{SAMPLE INTERVAL (SECONDS)}$

$H = 0.1$

$H = 0.2$

$H = 0.4$

FIGURE 21
The phase variance was also determined using a digital simulation to check the analytical results. In order to get worst case results the offset frequency of the input sine wave was made small (.1 rad/sec). For low offset frequencies adjacent samples of quantization noise are no longer independent causing the loop to be less effective in filtering out quantization noise. A graph of phase variance versus the number of quantization levels for both the theoretical and the digital simulation results is shown in Figure 22. As the number of levels is increased, adjacent samples of quantization noise become more decorrelated and the two curves approach each other. The interaction between thermal noise and quantization noise was investigated using a digital simulation to determine the output phase variance as a function of the number of quantization levels for different values of the output signal-to-noise ratio \( \frac{A^2}{2N_B} \). This graph, which is given in Figure 23, shows that the output phase variance is approximately independent of the number of quantization levels if the number of levels is greater than 16. Figure 23 shows that for 16 or more levels the standard deviation of the phase output is less than 1 degree for a wide range of signal-to-noise ratios. A phase error of this value in a PSK system would cause less than a 2 x \( 10^{-3} \) dB reduction in the effective output signal-to-noise ratio.

2.4.3 Scale Factor

Another important system constraint is the dynamic range of the A/D converter which precedes the digital computer. If the signal amplitude is greater than the dynamic range of the converter, signal energy will be lost resulting in a lower effective
STANDARD DEVIATION OF PHASE OUTPUT VERSUS THE NUMBER OF QUANTIZATION LEVELS

OFFSET FREQUENCY = 0.1 RAD/s

NUMBER OF LEVELS

STANDARD DEVIATION OF PHASE ERROR-DEGREES

DIGITAL SIMULATION

CALCULATED

FIGURE 22
PHASE ERROR VARIANCE VERSUS THE NUMBER OF QUANTIZATION LEVELS

$\sigma^2 = \frac{\text{output signal}}{\text{noise ratio}}$

Scale factor = 1.5

Number of quantization levels

Phase error variance $- \text{rad}^2$

$\sigma_0$ = 7 dB

$\sigma_0$ = 10 dB

$\sigma_0$ = 17 dB

FIGURE 23
input signal-to-noise ratio. If the dynamic range is made much greater than the
signal amplitude, the quantization noise will be increased because of the increased
size of the quantization interval. Therefore, the dynamic range of the converter
should be set somewhere between the above extremes. We determined the output
phase variance as a function of the number of quantization intervals for several
different dynamic range settings. These results showed that the output phase
variance was independent of the dynamic range if it was greater than \( (A = 3\sigma) \),
where \( A \) was the input sine wave amplitude, and \( \sigma \) is the input noise variance. A
graph of the output phase variance versus the number of quantization levels as
a function of the output signal-to-noise ratio and the dynamic range of the A/D
converter is shown in Figure 24. This curve shows that the phase variance is
reduced for a low number of quantization intervals as the dynamic range approaches
the input signal amplitude. Our results indicate that a dynamic range of \( (A + 2\sigma) \)
is a good compromise value.
Figure 24

**Scale Factor Determination**

- $(S/N)_o = \text{output signal - to-noise ratio}$
- $A = \text{input signal amplitude}$
- $\sigma = \text{input noise variance}$

<table>
<thead>
<tr>
<th>$(S/N)_o$</th>
<th>7 dB</th>
<th>10 dB</th>
<th>17 dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>A + $\sigma$</td>
<td>$A + 3\sigma$</td>
<td>$A + 3\sigma$</td>
<td>$A + 3\sigma$</td>
</tr>
</tbody>
</table>

Phase error variance - $\text{rad}^2$

Number of quantization levels

---

McDonnell Douglas Astronautics Company - East
2.5 Automatic Gain Control (AGC)

Two AGC techniques were investigated. The first approach consisted of placing a bandpass limiter at the input to the phase-locked loop. We also investigated a slight variation of this method, using a sawtooth comparator in combination with the bandpass limiter. The second technique considered was to use a closed loop AGC preceding the phase-locked loop. Both of these techniques are analyzed in Sections 2.5.1 and 2.5.2, and computer implementations are determined.

2.5.1 Bandpass Limiter

A block diagram of the phase-locked loop preceded by a bandpass limiter is given in Figure 25.

![Diagram of phase-locked loop with input limiter](image)

Assuming that the input signal is a narrow band process and that the higher harmonics will be removed by the bandpass filter following the limiter, the phase locked loop input will have the form given in Equation (58).

\[ V = \sin (\omega_0 t - \tan^{-1} \frac{Y}{X}) \]  

(58)

The error signal at the input to the filter is given in Equations (59) and (60). It should be noted that the 1/2 factor resulting is omitted and considered part of the loop gain.
The loop equation is then determined as shown in Equations (61) and (62).

\[
\frac{d}{dt} \left( \frac{1}{K} \frac{d\theta}{dt} \right) = (\frac{d}{dt} + a)e \tag{61}
\]

\[
\frac{d^2\theta}{dt^2} = aKe + K \frac{de}{dt} \tag{62}
\]

To avoid taking the derivative of the input signal, the above equation can be written as two first order equations by making the substitution \( \frac{dc}{dt} = aKe \).

\[
\frac{dc}{dt} = aK \left[ \frac{Y}{\sqrt{X^2 + Y^2}} \cos \theta - \frac{X}{\sqrt{X^2 + Y^2}} \sin \theta \right] \tag{63}
\]

\[
\frac{d\theta}{dt} = c + K \frac{Y}{\sqrt{X^2 + Y^2}} \cos \theta - \frac{X}{\sqrt{X^2 + Y^2}} \sin \theta \tag{64}
\]

Using Euler's Method, the difference equation for numerically solving the above equations will have the form given in Equations (65-67).

\[
G = \frac{1}{\sqrt{X_{n-1}^2 + Y_{n-1}^2}} \left[ Y_{n-1} \cos \theta_{n-1} - X_{n-1} \sin \theta_{n-1} \right] \tag{65}
\]

\[
\theta_n = \theta_{n-1} + hC_{n-1} + KhG \tag{66}
\]

\[
C_n = C_{n-1} + aKhG \tag{67}
\]

Using the Equations (65-67), a Fortran computer program was written for Euler's method and is given below.

- \( H = h = \) sample interval
- \( AK = Kh \)
- \( AA = a \)
- \( T = \) phase output of loop
XXA = ADC(01)
YYA = ADC(02)
XNOR = SQRT(XXA**2 + YYA**2)
XA = XXA/XNOR
YA = YYA/XNOR

TEMP = AK*(YA*COS(T) - XA*SIN(T))
T = T + C*H + TEMP
C = C + AA* TEMP

A slight variation on the preceding technique consists of using a bandpass
limiter preceding the phase-locked loop and a sawtooth comparator (inverse
sine circuit) in front of the loop filter. The resulting block diagram is
presented in Figure 26.

Using the same methods as were employed for the limiter, Equations (68) and
(69) were derived.

\[ \frac{dc}{dt} = aK \left( \tan^{-1} \frac{Y}{X} - \theta \right) \]  \hspace{1cm} (68)

\[ \frac{d\theta}{dt} = c + K \left( \tan^{-1} \frac{Y}{X} - \theta \right) \]  \hspace{1cm} (69)
A Fortran computer implementation of this technique is given below.

\[ H = h = \text{sample interval} \]
\[ AK = Kh \]
\[ AA = a \]
\[ T = \text{phase output of loop} \]
\[ XXA = \text{ADC}(01) \]
\[ YYA = \text{ADC}(01) \]
\[ \text{ANG2} = \text{ATAN2}(YYA, XXA) \]
\[ \text{ANG2} = \text{ASIN}(\sin(\text{ANG2}-\text{T})) \]
\[ \text{TEMP} = AK \times \text{ANG2} \]
\[ T = T + C \times H + \text{TEMP} \]
\[ C = C + AA \times \text{TEMP} \]

A curve showing the output phase variance as a function of the input noise spectral density for a phase-locked loop with, and without, a limiter is shown in Figure 27. This curve shows that as the input noise spectral density increases, the limiter reduces the gain of the loop, and as a result the output phase variance is reduced with respect to the loop with perfect AGC. A similar curve for the sawtooth comparator is shown in Figure 28. Since the sawtooth comparator exhibits more gain for large phase offsets than a loop with a sinusoidal comparator and an input limiter, the phase variance is greater for the sawtooth comparator configuration as low input signal-to-noise ratios. It should also be noted that the sawtooth comparator will track the input variations more effectively than the techniques described previously.

2.5.2 Closed Loop AGC

A block diagram for an analog configuration of a closed loop AGC is given in Figure 29.
PHASE ERROR VARIANCE FOR LIMITER

SAMPLE TIME = 0.2 s

PERFECT AGC

LIMITER

LINEAR APPROXIMATION

\( \frac{N_0 B}{A^2} \)

FIGURE 27
PHASE ERROR VARIANCE FOR SAWTOOTH COMPARATOR

SAMPLE INTERVAL = 0.2 s

PHASE ERROR VARIANCE = $\text{rad}^2$

$N_0B_w/A^2$

FIGURE 28
The incremental transfer function for this loop will have the form given in Equations (70) and (71) according to Povejsil, Raven, and Waterman.\(^3\)

\[
\frac{\Delta E_{\text{out}}}{\Delta E_{\text{in}}} = \frac{1}{1 + K_2 F(s)} \tag{70}
\]

\[
K_2 = \left. \frac{dE_{\text{out}}}{de} \right|_{E_{\text{in}} = \text{constant}} \tag{71}
\]

In order to keep the loop gain independent of the input level, an exponential gain characteristic was used. An integrator was chosen for the loop filter to reduce the steady-state offset between \(E_{\text{out}}\) and \(V_{\text{ref}}\) to zero. The resulting loop transfer characteristic is given in Equations (72-75).

\[
\frac{\Delta E_{\text{out}}}{\Delta E_{\text{in}}} = \frac{s}{s + K_2} \tag{72}
\]

\[
E_{\text{out}} = G(e_g) E_{\text{in}} \tag{73}
\]

\[
G(e_g) = e^{Be_g} \tag{74}
\]

\[
\left. \frac{\partial E_{\text{out}}}{\partial e_g} \right|_{E_{\text{in}} = \text{constant}} = K_2 = E_{\text{in}} Be_g \tag{75}
\]
A block diagram of the digital implementation of this circuit is shown below in Figure 30.

![IMPLEMENTATION DIAGRAM FOR AGC LOOP](image)

A Fortran computer program for implementing the above technique is given below.

\[ H = h = \text{sample interval} \]
\[ \text{GAIN} = \frac{K}{V_{\text{ref}}} \]
\[ AK = Kh \]
\[ AA = a \]
\[ T = \text{phase output of loop} \]
\[ \text{REF} = V_{\text{ref}} \]
\[ X = G \times \text{ADC(01)} \]
\[ Y = G \times \text{ADC(02)} \]
\[ \text{ENV} = \sqrt{X^2 + Y^2} \]
\[ \text{ERR} = \text{ENV} - \text{REF} \]
\[ \text{EIT} = \text{EITM} + \text{ERR} \times H \]
\[ \text{EITM} = \text{EIT} \]
\[
G = \exp(-\text{GAIN} \times \text{EIT})
\]
\[
\text{TEMP} = \text{AK} \times (X \times \cos(T) - Y \times \sin(T))
\]
\[
T = T + C \times H + \text{TEMP}
\]
\[
C = C + \text{AA} \times \text{TEMP}
\]

2.5.3 Comparison of Techniques

In order to show that the two AGC implementations given above performed properly, we tested each technique by jittering the amplitude of the input signal and measuring the variance in the gain of the phase-locked loop. For a 5% input amplitude variance the gain of the loop varied by 3.4% for the closed loop configuration and 1.1% for the limiter. This result would be expected since the limiter has an effective time constant which is infinitely small, while the closed loop configuration that was investigated had a time constant of 5 seconds. The operation of the AGC circuit was also demonstrated by showing the effect of the input signal to noise ratio on the tracking ability of a phase-locked loop preceded by one of the above AGC circuits. Utilizing a digital simulation, a sine wave phase input was applied to the loop, and the phase error variance was determined as a function of the input signal-to-noise ratio. A graph of these results is shown in Figure 31. For input signal-to-noise ratios above zero dB the output phase variance approaches the theoretical value for white noise passed through a linear loop. This indicates that the phase-locked loop is tracking the input sinusoidal phase variation with negligible error. As the input signal-to-noise ratio is reduced, the loop gain is reduced and the input phase variations can no longer be tracked. Figure 31 indicates that the threshold occurs at an input signal-to-noise ratio of -4 dB for the limiter and -7 dB for the closed loop AGC with a 5 second time constant. Therefore, although the limiter
AGC reduces the gain variations by more than the closed loop configuration, it has a higher threshold than the latter method.

These results indicate that the limiter gives better regulation than the closed loop AGC, but it has a higher threshold. The previous Fortran computer programs showed that the limiter type AGC is faster and less complex than the closed loop approach. Since speed and complexity are important considerations in implementing a digitized receiver, the limiter AGC was chosen as the optimum method of obtaining automatic gain control.

**PHASE ERROR VARIANCE VERSUS INPUT SNR**

\[
\sigma^2 \theta = \frac{N_0 B_n}{A^2} = 0.1
\]

- \(B_n\) = NOISE BANDWIDTH
- \(N_0\) = SINGLE SIDED INPUT NOISE SPECTRAL DENSITY
- \(r\) = CLOSED LOOP AGC TIME CONSTANT

**FIGURE 31**

**CLOSED LOOP \((r = 5s)\)**

**LIMITER**
2.6 **Acquisition**

We analyzed and compared three approaches to phase-locked loop acquisition; the swept frequency method, the Fast Fourier Transform method, and the free-running mode. The first technique consists of beating the input signal with a swept frequency oscillator and passing the resultant signal through a low pass filter and a threshold detector. The second approach consists of computing the FFT of N input samples and then determining the Fourier coefficient of maximum amplitude. The final method allows the phase-locked loop to pull-in with no external controls.

2.6.1 **Swept Frequency Method**

The first technique, which was studied by Sterling, consists of beating the input signal with two oscillators that are 90° out of phase, filtering the resultant signal, and performing a threshold detection to determine when the swept frequency equals the input frequency. The two out-of-phase oscillators are necessary since the phase of the input signal is not known and thus a zero output could result if only one oscillator was used. A block diagram of this configuration is shown in Figure 32. The input multiplications shown in this figure are derived in Equations (76-81).

\[
E_i = X \sin \omega_0 t + Y \cos \omega_0 t \quad (76)
\]

\[
E_{vco} = \cos (\omega_0 t + \theta) \quad (77)
\]

\[
\theta = \omega_0 t - kt^2 \quad (78)
\]

\[
k = \frac{\omega_0}{T} \quad (79)
\]

\[
T = \text{sweep time}
\]

\[
e_i = E_i E_{vco} = -X \sin \theta + Y \cos \theta \quad (80)
\]

\[
e_q = E_i E_{vco} = X \cos \theta + Y \sin \theta \quad (81)
\]
FIGURE 32
Each of the quadrature components is passed through a low pass filter and an absolute value circuit. The resultant sum signal is applied to a threshold detector. The frequency of the sweep at the time the threshold is crossed is used as the initial condition on the frequency of the phase-locked loop. If this frequency is within the pull-in range of the phase-locked loop, the loop will almost instantaneously acquire the signal. The time constant of the low pass filter was chosen to maximize the ratio between the peak signal at the input to the threshold detector and the standard deviation of the noise at the output of the low pass filter. The measurement of peak signal to rms noise at the filter output is only an approximation of the actual signal-to-noise ratio since the noise should be taken at the input to the threshold detector. However, the absolute value circuit makes this difficult, and thus the approximate measurement was made. A graph of the signal-to-noise ratio versus the ratio of the low pass filter bandwidth squared to the sweep rate is shown in Figure 33. This graph shows that the signal-to-noise ratio is maximized for \( \frac{\omega^2_{LP}}{2K} \) equal to 0.1715. However, this ratio can be varied between 0.1 and 0.275 with a resulting loss in signal-to-noise ratio of less than 0.1 dB. A computer program for the above acquisition circuitry with the phase-locked loop included is shown in Figure 34. When the variable E is greater than the threshold, the variable C is set equal to the swept frequency at that time, and the phase-locked loop calculations are begun at statement 2.

2.6.2 Fast Fourier Transform Acquisition

A second method for performing phase-locked loop acquisition is to utilize the Fast Fourier Transform (FFT). This technique consists of reading N samples into the computer and calculating the Discrete Fourier
Transform (DFT) which is given in Equation (82).

\[ A_r = \sum_{k=0}^{N-1} X_k \exp(-2\pi j r k/N) \]  
\[ r = 0, 1, ..., N-1 \]

\( X_k \) = input samples

Determining Relationship Between LPF Bandwidth and Sweep Rate

\( \omega_{lpf} \) = LOW PASS FILTER BANDWIDTH - RAD/s
\( 2K \) = SWEEP RATE - RAD/s^2

**FIGURE 33**

**MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST**

53
The FFT algorithm removes redundant operations from the calculations and reduces the number of operations from $N^2$ to $2N \log_2 N$. For large values of $N$ this is an extremely significant reduction. The listing of a digital computer program for implementing the FFT algorithm is given in Figure 35.
DIGITAL COMPUTER PROGRAM FOR FFT GENERATION

C COMPLEX W, X, Y
C N = NUMBER OF SAMPLES
C N = 2**L
C X(J) = INPUT DATA
PI = 3.141592653589793
BBN = N
AR = 2.* PI/BBN
MM = 0
K = 1
NA = N/2
KK = NA
DO 2 J = 1, L
IF (MM .EQ. 1) GO TO 9
MM = 1
DO 1 I = 1, K
IM = (I - 1) * KK
AA = IM*AR
W = CMPLX (COS (AA), - SIN (AA))
DO 11 I = 1, KK
NZ = I + IM
NQ = NZ + IM
Y(NZ) = X(NQ) + W*Y(NQ + KK)
Y(NZ + NA) = Y(NQ) - W*Y(NQ + KK)
1 CONTINUE
GO TO 12
9 MM = 0
DO 10 I = 1, K
IM = (I - 1) * KK
AA = IM*AR
W = CMPLX (COS (AA), - SIN (AA))
DO 11 I = 1, KK
NZ = I + IM
NQ = NZ + IM
X(NZ) = Y(NQ) + W*Y(NQ + KK)
X(NZ + NA) = Y(NQ) - W*Y(NQ + KK)
10 CONTINUE
12 KK = KK/2
K = 2*K
2 CONTINUE
C X(J) = OUTPUT DATA IF L IS EVEN
C Y(J) = OUTPUT DATA IF L IS ODD

FIGURE 35

The program, which was written in Fortran IV, determines the FFT of a sequence of N complex input samples X(K). This program uses complex input samples which
is of great advantage since the Fourier Transform of the IF signal \((X \cos \omega_0 t + Y \sin \omega_0 t)\) can be obtained by frequency shifting the transform of \((X + jY)\) as is shown in Equations (83) and (84).

\[
F(X + jY) = F(\omega)
\]

\[
F(X \cos \omega_0 t + Y \sin \omega_0 t) = \frac{1}{2} F^*(\omega - \omega_0) + \frac{1}{2} F(\omega + \omega_0)
\]

The sampling rate must be greater than twice the maximum frequency of the sampled signal. The total number of samples determines the separation between the discrete spectral components. The program shown in Figure 35 has been written such that \(N\) must be an integral power of two \((N = 2^L)\). This constraint is not usually restrictive in using this program for acquisition since the number of samples can usually be increased such that \(N\) becomes an integral power of two. The preceding program stores the DFT of the input signal in either the X or Y matrix depending on whether \(L\) is even or odd. The relationship between the coefficients \(A_r\) and the elements of the output matrix (assumed to be X for this example) is given below.

\[A_r = X(r + 1)\]

If the function \((X + jY)\) is transformed, where \(X\) and \(Y\) are the quadrature components of the input signal, the complex value of the various spectral components are found in the storage locations given in Equation (85).

\[
f_1 = \frac{1}{Nh}
\]

\(h = \text{sample interval}
\)

\(N = \text{number of samples}
\)

\[X(1) \rightarrow f_0 \quad X(N) \rightarrow f_0 - f_1\]

\[X(2) \rightarrow f_0 + f_1 \quad X(N-1) \rightarrow f_0 - 2f_1\]
After computing and storing the complex Fourier coefficients, the magnitude of each coefficient is determined. The estimate of the input frequency is then made by determining the frequency associated with the complex Fourier coefficient of maximum magnitude. This value of frequency is then inserted as an initial condition in the the phase-locked loop as was done with the swept frequency method.

2.6.3 Free-Running Acquisition

This technique allows the phase-locked loop to pull-in without any external control. If the initial frequency offset is within the pull-in range of the loop, acquisition will occur. However, for large frequency offsets the loop will begin slipping cycles, and pull-in will occur in a much longer time. To show this effect we measured the mean and variance in pull-in time as a function of the frequency offset between the VCO and the input sine wave and the output signal-to-noise ratio. This investigation showed that the average pull-in time was independent of the output signal-to-noise ratio. A graph of pull-in time versus frequency offset is shown in Figure 36. The variance in pull-in time, which is affected by the output signal-to-noise ratio, is graphed in Figure 37 as a function of the frequency offset.

2.6.4 Comparison of Techniques

The free-running acquisition approach was discarded because of the excessive pull-in time required for large frequency offsets. The two remaining techniques were compared using digital simulation. The probability of acquisition was determined as a function of the sweep rate for two different output signal-to-noise ratios. In the simulation a correct acquisition was assumed to have
MEAN PULL-IN TIME VERSUS OFFSET FREQUENCY

OFFSET FREQUENCY - RAD/s

PULL IN TIME - s

FIGURE 36

58

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
Variance in pull-in time versus frequency offset

\((S/N)_0 = \text{output signal to noise ratio}\)

\((S/N)_0 = 7 \text{ dB}\)

\((S/N)_0 = 10 \text{ dB}\)

occurred if the difference between the estimate of the input frequency and the input frequency was within the pull-in range of the phase-locked loop. The loop pull-in range is considered to be the range over which the loop will acquire without slipping cycles. The results of this investigation are shown in Figure 38.

The sweep rate for the FFT technique is determined by the frequency range, the sample interval, and the number of samples. Figure 38 indicates that the FFT
technique gives much better results than the sweep control method. This is due to the fact that the FFT method is similar to a parallel bank of matched filters centered on each of the spectral lines, while the latter method approximates a single matched filter which is switched from one spectral line to another. Therefore, the integration time is much longer for the FFT technique and the output noise power is reduced.

The main criteria used to choose between the acquisition techniques is speed and simplicity. One of the big disadvantages of the FFT technique is that it must be done sequentially. This means that no calculations can be made until all N samples are read into the digital computer, and that the digital
processor must be capable of storing these N samples. These constraints
decrease the speed and increase the complexity associated with the processor.
The calculations associated with the sweep control method can be made between
samples and thus fewer modifications are necessary with this technique.
Therefore, the sweep control method is the recommended approach for solving
the acquisition problem.

2.7 Hybrid Simulation

One of the important decisions associated with implementing a digital processor
is the determination of the filter which precedes the A/D converter. We first
determined the effect of three candidate input filters, an integrate and dump filter,
a single break low pass filter, and a double break low pass filter, on system perfor-
mance. We then set up a hybrid simulation of the digitized phase-locked loop and
determined the phase error variance as a function of noise-to-signal ratio.

2.7.1 Input Filter

Each of the quadrature components is filtered before being passed into the digital
computer. This filter, which precedes the sample and hold circuit, eliminates the
second harmonic of the input signal and reduces the loss caused by aliasing. If it
is assumed that the input signal remains constant between samples, an integrate
and dump filter, which integrates during the time between samples, is the optimum
filter. The main problem associated with this technique is that a timing signal
must be sent from the digital computer to dump the external integrator at the
appropriate time. Because of the extra complexity associated with the above method,
two alternative techniques, a single break low pass filter and a two break low
pass filter, were considered. The transfer functions for these two filters are
given in Equations (86) and (87).

\[
F_1(s) = \frac{\alpha}{s + \alpha} \quad (86)
\]

\[
F_2(s) = \frac{\alpha^2}{(s + \alpha)^2} \quad (87)
\]
In order to compare the three different filter implementations, we assumed that the input filter bandwidth was wide enough that the effective noise spectral density of the sampled and held input signal is constant over the noise bandwidth of the phase-locked loop. The loss in input signal to noise ratio is then defined as given in Equation (88),

\[ L = 10 \log_{10} \left( \frac{S_n(0)}{N_0 F(w_{\text{max}})} \right)^2 \]  

(88)

where:

- \( L \) = loss in input signal-to-noise ratio (dB)
- \( \omega_{\text{max}} \) = maximum offset frequency
- \( F(\omega) \) = filter transfer function
- \( N_0 \) = noise spectral density at filter input
- \( S_n(0) \) = effective noise spectral density at the output of the sample and hold circuit

For the integrate and dump filter \( S_n(0) \) is equal to \( N_0 \). Therefore, the only loss results from the decrease in amplitude of the maximum offset sine wave at the output of the integrate and dump filter. This loss is then given in Equation (89).

\[ L = 20 \log_{10} \left( \frac{\sin \frac{\pi}{R}}{\pi/R} \right) \]  

(89)

where:

- \( R = \frac{f_s}{f_{\text{max}}} \)
- \( f_s \) = sampling rate
- \( f_{\text{max}} \) = maximum offset frequency

The increase in noise spectral density at zero frequency for the input filter \( F(s) \) was determined from Equation (90).

\[ \frac{S_n(0)}{N_0} = \frac{1}{2} + \sum_{n=1}^{\infty} F(nw_s)^2 \]  

(90)
This formula is determined from the fact that the spectral density at the input to a sampler is reproduced at multiples of the sampling frequency at the output of the sampler. Using the above relationships, the loss for filters $F_1(s)$ and $F_2(s)$ is given in Equations (91-94).

$$L_1 = 10 \log_{10} \left[ \frac{\frac{1}{2} + \sum_{n=1}^{\infty} \frac{1}{1 + (nk)^2}}{\left( \frac{R^2}{R^2 + K^2} \right)^2} \right]$$

(91)

$$L_2 = 10 \log_{10} \left[ \frac{\frac{1}{2} + \sum_{n=1}^{\infty} \left( \frac{1}{1 + (nk)^2} \right)^2}{\left( \frac{R^2}{R^2 + K^2} \right)^2} \right]$$

(92)

$$K = \frac{f_s}{f_{LPF}}$$

(93)

$$R = \frac{f}{f_{max}}$$

(94)

In order to minimize the loss for each of the two filter configurations, we selected a value for $R$ and plotted $L$ as a function of $K$. One of these graphs for the second order filter for $R=10$ is shown in Figure 39. This curve shows that the minimum value of $L_2$ is .683 dB and that it occurs at $K = 2.26$. Utilizing the digital computer, we repeated the above procedure for different values of $R$. A graph of the minimum value of the loss versus $R$ is shown in Figure 40 for all three filter configurations. Figure 41 shows a plot of the optimum value of $K$ versus $R$. The integrate and dump circuit is not shown in this final figure since the ratio of sampling rate to filter bandwidth is a constant. The above results show that the integrate and dump filter is 1/2 dB better than a second order filter and 1 1/4 dB better than the first order filter for values of $R$ equal to ten.
OPTIMIZING SECOND ORDER INPUT FILTER

Figure 39

Loss in Input SNR - dB

$\frac{f_s}{f_{lp}}$

$\frac{f_s}{f_{lp}}$ MAX = 10.0
COMPARING INPUT FILTERS (L VERSUS R)

- SINGLE BREAK FILTER
- DOUBLE BREAK FILTER
- INTEGRATE AND DUMP

FIGURE 40
2.7.2 Results

Utilizing our 1 MHz "front end," the adage 770 A/D converter, and the CDC 6400 computer, a hybrid simulation of the digital phase-locked loop was conducted.

A simplified block diagram of the configuration used is given in Figure 42.
Figure 43 shows strip chart recordings of the input quadrature components and the phase output of the loop which was fed back from the digital computer. The phase error variance was then determined by generating an internal phase reference using a second phase-locked loop with a bandwidth 300 times smaller than the first. The only restriction on this technique is that the second loop must be allowed to run for a sufficient length of time before the phase error calculation begins so that any initial transient has decayed off. Another method of generating a phase reference would be to pass the non-noisy signal through an identical "front end" and compute the phase in the digital computer. This approach was discarded because of the problems associated with balancing the two channels and the added complexity associated with adding the two extra channels. Figures 44 and 45 show the phase error variance versus $\frac{N \cdot B}{A^2}$ for the phase-locked loop without a limiter and with a limiter respectively.

For both of these simulations the input filter consisted of a single break low pass filter with a 45 Hz bandwidth. We also set a .2Hz offset between the input frequency and the IF center frequency.
FIGURE 43
PHASE ERROR VARIANCE WITHOUT LIMITER
(HYBRID SIMULATION)

- $h = 5\text{ ms}$
- $B_n = 30\text{ Hz}$
- $\zeta = 0.707$

**DIGITAL SIMULATION**

**LINEAR APPROXIMATION**

$\frac{N_0 B_n}{A^2}$ vs. PHASE ERROR VARIANCE - RAD$^2$

**FIGURE 44**
2.8 Evaluation of Performance

In this section we compare the results obtained during this study on phase error variance, phase-locked loop threshold, and acquisition with the results obtained by other investigators.
2.8.1 Phase Error Variance

In order to validate the results obtained during this portion of the study, we determined the relationship between the phase error variance and the noise-to-signal ratio using both an approximate technique, quasi-linearization, and an exact approach, the Fokker-Plank Method. Quasi-linearization consists of replacing the nonlinearity with an equivalent gain which is determined from a knowledge of the statistics at the input to the nonlinearity. If it is assumed that the input to the sine nonlinearity in the phase-locked loop is Gaussian with a variance $\sigma^2_\phi$, Develet\(^5\) shows that the equivalent gain can be determined as given in Equations (95) and (96).

\[
G_{\text{EQ}} = \frac{AK}{\sqrt{2\pi} \sigma_\phi} \int_{-\infty}^{\infty} e^{-\theta^2/2\sigma_\phi^2} \cos \theta \, d\theta \quad (95)
\]

\[
G_{\text{EQ}} = AK e^{-\sigma_\phi^2/2} \quad (96)
\]

$A =$ input amplitude

$K =$ VCO gain

This value of gain is substituted into the equation for phase variance at the output of the linear loop.

\[
\frac{A^2}{N_B} = \frac{e^{\sigma_\phi^2/2}}{\sigma^2_\phi} \left[ \frac{1 + 4\zeta^2 e^{-\sigma_\phi^2/2}}{1 + 4\zeta^2} \right] \quad (97)
\]

$B_n =$ noise bandwidth of loop for $\sigma_\phi^2 = 0$

$\zeta =$ damping ratio of loop

A graph of this equation is given in Figure 46 along with the results obtained using our digital simulation. The maximum difference between the digital simulation results and the quasi-linear approximation is about 20%. As the phase error increases,
the Gaussian approximation no longer holds, and thus we would expect the curves to deviate as they do. A better method of validating the simulation results is to solve the Fokker-Plank equation of the phase-locked loop for the probability density of the phase error. The exact value of the phase error can then be determined as
a function of \( \frac{N B_n}{A^2} \). However, no exact solution has been obtained for a second
order loop with the filter \( F(s) = \frac{s+a}{s} \). Therefore, we digitally simulated a phase-
locked loop with the filter \( F(s) = \frac{a}{s+a} \) for which a solution to the Fokker-Plank
Equation can be obtained. Viterbi\textsuperscript{6} and Lindsey and Tauseworthe\textsuperscript{7} show that the
probability density function of the phase error and the phase error variance will
have the form given in Equations (98 - 100).

\[
p(\phi) = \frac{\alpha \cos \phi}{2\pi I_0(\alpha)} \quad -\pi < \phi < \pi \tag{98}
\]

\[
\alpha = \frac{A^2}{N R_n} \tag{99}
\]

\[
\sigma_\phi^2 = \frac{\pi^2}{3} + 4 \sum_{n=1}^{\infty} \frac{(-1)^n \ln(\alpha)}{n I_0(\alpha)} \tag{100}
\]

A graph of \( \sigma_\phi^2 \) versus \( \frac{1}{\alpha} \) is shown in Figure 47. This plot shows that there
is negligible error between the calculated and experimental results for a sample
time of .05 seconds. For slower sampling rates, the curves will deviate slightly
because of errors associated with the numerical technique. For \( \alpha = 2 \) the error
between the two curves was 15% for a .1 second sampling interval and 30% for
a .2 second interval.

2.8.2 Phase-Locked Loop Threshold

There are several different ways of defining the phase-locked loop threshold.
The method employed by Viterbi\textsuperscript{6} is to develop a model and then determine when
the actual phase variance deviates by a specified amount from that predicted
by the given model. In this section we will arbitrarily select the linear model
as our standard and assume that the threshold occurs at the point where the
error equals 50%. Using this criterion, the threshold noise-to-signal ratio,
\( \frac{N B_n}{A^2} \), is given for several different loop configurations and sampling
intervals, \( h \).
Comparing Digital Simulation Results with Fokker-Plank Solution

\[ \zeta = 0.707 \]
\[ \omega_n = 1.414 \text{ RAD/s} \]
\[ h = 0.05 \text{ s} \]
\[ F(s) = \frac{2}{s + 2} \]

Figure 47
These results show that the threshold signal-to-noise ratio is increased as the sampling rate is increased. We have already shown that the results obtained with a sample interval of .05 seconds closely approximate the results obtained for an analog loop. For a .2 second sample interval, the threshold is increased by 3 db in going from the analog to the digitized loop.

2.8.3 Acquisition

We measured acquisition time as a function of the initial frequency offset and the results are presented in Section 2.6. Viterbi\(^8\) determined an approximate formula for acquisition time for the second order phase-locked loop with filter \(F(s) = \frac{s + a}{a}\), which is given in Equation (101).

\[
t = \frac{1}{a} \left\{ \left[ \frac{1}{AK} \frac{d\phi(o)}{dt} \right]^2 - \dot{\phi}_A^2 \right\} \tag{101}
\]

\(AK = \) loop gain
\(a = \) filter constant
\(\dot{\phi}_A = \) limit of frequency lock \(= f \left( \frac{a}{AK} \right)\)

The above approximation becomes increasingly crude as \(\left[ \frac{1}{AK} \frac{d\phi(o)}{dt} \right]\) approaches \(\dot{\phi}_A\). A graph of this function is given in Figure 48 with the experimentally measured points superimposed. These results indicate that there is negligible difference between the digital simulation results and the theoretical values.
COMPARING MEASURED AND CALCULATED PULL-IN TIME

FIGURE 48
We also measured the probability of acquisition using a digital simulation of a system using a swept VCO. This technique is different from the one previously mentioned (Section 2.6.1) because the VCO sweep is not stopped when it is within the pull-in range of the loop. The loop must actually begin tracking the phase input which consists of a ramp of frequency. A graph of the results of this investigation is given in Figure 49 for an output signal-to-noise ratio of 10 dB. Also plotted on this graph are experimental results obtained by Frazier and Page for an analog loop. They used a second order loop with a lead-lag filter of the form \( F(s) = \frac{s+a}{s+b} \). However, since \( (a) \) is 1,000 times \( (b) \), this configuration...
closely approximates that used in the digital simulation. The maximum error between the results for the digitized and the analog loops is 17%. This error is not significant when we consider the difficulty in determining whether acquisition has taken place and the fact that both results were experimentally obtained. The results for the digitized loop would also be closer to the analog results if the sampling rate was increased.
3. DEVELOPMENT OF DIGITIZED PSK RECEIVER

3.1 Synthesis of PSK Receiver Software

In Section 2, we studied techniques for synthesizing digital subsystems in general, and the phase-locked loop in particular. We applied these results in Section 3 to the development of a completely digital PSK receiver. Various solutions to the carrier and bit synchronization problem are compared on the basis of performance, speed, and complexity.

3.1.1 Sampling and A/D Conversion of Data

The first step in digital receiver design is the specification of a sampling technique. A brute force method would be to sample the noisy carrier at a rate which is fast enough to reproduce the carrier frequency. Digital logic and A/D converters, however, are not capable of operating at speeds compatible with typical carrier frequencies. For this reason the utility of the quadrature component signal sampling technique developed earlier in this report is of special importance in digital receiver design for narrowband binary PSK signals. With this technique the RF carrier is first heterodyned to a convenient IF frequency. There are two approaches to obtaining the quadrature components from the IF.

Quadrature Sampling

In one approach the quadrature components of the PSK carrier are generated by heterodyning the carrier with the best available estimate of the carrier and with a 90° phase shifted version of the carrier. Both of these products are then low pass filtered to eliminate the double-frequency terms (Figure 50a).
The output of these filters are the quadrature components of the noisy PSK signal. Note that it is only necessary to sample the quadrature components at a rate consistent with the signal bandwidth. The technique is very efficient since it reduces the bandwidth and sampling rate to levels consistent with the information bandwidth and places the minimum possible demands on the A/D converter time resolution.

**IF Sampling**

This section describes the direct IF sampling approach (Figure 50b) and the usage of the ADCOM IF Sampler with the MDAC digital receiver.

The direct IF sampling technique generates a pair of quadrature samples by strobing a narrow aperture sampler at two instants of time separated by one-quarter of a period of the nominal IF (Figure 51). Denote the noisy PSK
signal referred to the nominal IF frequency, \( \omega_o \), by Equation (102)

\[
E(t) = A \, m(t) \sin \left( \omega_o t + \theta(t) \right) + n(t) \tag{102}
\]

\[
n(t) = n_1(t) \cos \left( \omega_o t + \theta(t) \right) + n_2(t) \sin \left( \omega_o t + \theta(t) \right) \tag{103}
\]

\( m(t) = \text{modulation} \)

In terms of its quadrature components, \( x(t) \) and \( y(t) \) defined in Equation (104) and Equation (105), the noisy signal can be represented as Equation (106).

\[
x(t) = \left[ A \, m(t) + n_2(t) \right] \cos \theta(t) - n_1(t) \sin \theta(t) \tag{104}
\]

\[
y(t) = \left[ A \, m(t) + n_2(t) \right] \sin \theta(t) + n_1(t) \cos \theta(t) \tag{105}
\]

\[
E(t) = x(t) \sin \omega_o t + y(t) \cos \omega_o t \tag{106}
\]

The first quadrature sample taken at \( t_1 = N \frac{2\pi}{\omega_o} \) is shown in Equation (107).

\[
a_1 = E(t_1) = y(t_1) \tag{107}
\]

The second quadrature sample taken at \( t_1 + \frac{\pi}{2\omega_o} \) is shown in Equation (108).

\[
a_2 = E \left( t_1 + \frac{\pi}{2\omega_o} \right) = x \left( t_1 + \frac{\pi}{2\omega_o} \right) \tag{108}
\]

\[
a_2 \approx x(t_1) \tag{109}
\]

Equation (109) follows because \( m(t) \), \( \theta(t) \), \( n_1(t) \), and \( n_2(t) \) are slowly varying compared to \( \omega_o \). Thus, \( a_1 \) and \( a_2 \) are the same quadrature samples that are obtained from the quadrature sampling technique.

The ADCOM IF Sampler is operable at three IF frequencies, 10 MHz, 1 MHz, and 1 KHz. For 1 KHz inputs the required 90° separation is achieved by operating each sample and hold with a repetition period of 256 μsec. The output of each sample and hold corresponds to samples spaced by 90° (i.e., 0°, 90°, 180°, 270°, ...) of an IF reference at \( (4 \times 256)^{-1} \times 10^6 = 0.97 \text{ KHz} \) (Figure 52).
The digital processor can change the effective sampling rate by discarding certain samples. For example, the first two samples from S/H 1 in Figure 52 are quadrature samples of 0.97 KHz IF. The digital processor fixes the interval between input samples of the quadrature components at 2.048 msec by discarding six samples from S/H 1 before accepting the ninth and tenth samples as the next pair of quadrature component samples.

The standard test parameters for the MDAC digital receiver were chosen to be a 244 Hz IF with a bit rate of 12.2 bits/second and a sample period of \( \frac{1}{244} \) seconds (20 samples/bit). The ADCOM IF Sampler was adjusted so that the S/H's operate with a 1.024 msec period. The digital computer accepts four samples from the output of one sample and hold operating with a sample period of 1.024 msec. The first two samples are the quadrature components of the 244 Hz input signal (Figure 53). The remaining two samples are discarded.
For this sampling technique, there is one carrier cycle between successive quadrature sample pairs. Sampling with a noninteger number of carrier cycles between quadrature sample pairs introduces an apparent frequency offset at \( \frac{d}{N+d} \omega_o \) rad/sec where \( N+d \) is the total number of carrier cycles between quadrature sample pairs. \( N \) is an integer and \( d \) is less than unity. The input signal to the digital receiver is then described by Equation (110) instead of Equation (102).

\[
E(t) = A m(t) \sin \left[ \omega_o t + \theta(t) + \frac{d}{N+d} \omega_o t \right] + n(t) \tag{110}
\]

When operating the MDAC digital receiver with a noninteger number of carrier cycles between sample pairs a frequency offset, \( \frac{d}{N+d} \omega_o \) rad/sec is used as an initial condition in the Costas loop.

3.1.2 Rejection of DC Bias

In experimental evaluations of the digital receiver, we experienced difficulties with a DC bias at the output of the quadrature component generation circuitry. This bias appears to the Costas loop to be an interfering frequency component at the center frequency of the IF filter. If the bias is
large enough, the Costas loop will track the IF center frequency rather than the input signal frequency. We investigated both placing an analog high pass filter in the quadrature component circuitry and placing a digital high pass filter in the digital receiver. The most important trade-off between these techniques is the extra computation time associated with the digital implementation as opposed to the requirement of an added analog filter.

We first incorporated a digital high pass filter into the digital receiver as shown in Figure 54. Any DC bias in the quadrature components is eliminated

\[ H(f) = \frac{1}{1 + j2\pi fr} \]

**DC BIAS REMOVER**

Figure 54
by first passing each quadrature component through a digital low pass filter and then subtracting the low pass filter output from the quadrature component.

The Z transform implementation for a low pass filter whose transfer function is $H(f)$ as defined in Figure 54 is given in Equation (111) where $E_0(n)$ is the $n$th input sample, $E_o(n)$ is the $n$th filter output, and $h$ is the sample period.

$$E_o(n) = e^{-\frac{h}{\tau}} E_o(n-1) + (1 - e^{-\frac{h}{\tau}}) E_i(n-1)$$

(111)

for $\frac{h}{\tau} \ll 1$, $e^{-\frac{h}{\tau}} \approx 1 - \frac{h}{\tau}$

$$E_o(n) = E_o(n-1) - \frac{h}{\tau} E_o(n-1) + \frac{h}{\tau} E_i(n-1)$$

(112)

Equation (112) uses only two multiplications and two additions. One subtraction is then needed to remove the DC bias from the quadrature component. These operations do not add any significant computation time. We measured Costas loop tracking performance and phase error variance with the DC bias remover added and found no detrimental effect on receiver performance for $\tau = 200$ bit periods. The digital high pass filter is easier to implement than an analog high pass filter and does not significantly increase computation time. For these reasons we chose the digital implementation.

3.1.3 Input Filter Design

Section 2.7.1 discussed the loss due to sampling or noise aliasing and showed that it was desirable to make the ratio of sampling rate to input bandwidth as large as possible. The input bandwidth is fixed by the bit rate and frequency instability. If all the input samples are passed through the entire digital receiver, the maximum sampling rate is determined by the maximum required computation time. However, one way to increase the effective sampling rate is to sample at a high rate, filter a group of these samples to
produce one smoothed sample, and then process this one sample with the entire receiver. This is roughly equivalent to lowering the input filter bandwidth, but it is accomplished purely by digital means. The sampling loss is reduced due to the increase in the ratio of sampling rate to input bandwidth.

For instance, when operating at 244 Hz, the IF signal is sampled four times each carrier cycle (Figure 53). Normally, the first two samples are processed by the digital receiver and the remaining samples are discarded. In this section we use all of the input samples to obtain signal quadrature components.

Define the four samples in one sample interval as follows:

\[ a_1 = y(t_1) \]  \hspace{1cm} (113)

\[ a_2 = x(t_1 + \frac{\pi}{2\omega_0}) \approx x(t_1) \]  \hspace{1cm} (114)

\[ a_3 = y(t_1 + \frac{\pi}{\omega_0}) \]  \hspace{1cm} (115)

\[ a_3 = -y(t_1) \]  \hspace{1cm} (116)

\[ a_4 = x(t_1 + \frac{3\pi}{2\omega_0}) \]  \hspace{1cm} (117)

\[ a_4 = -x(t_1) \]  \hspace{1cm} (118)

Equations (116) and (118) follow because \( m(t), e(t), n_1(t), \) and \( n_2(t) \) are slowly varying compared \( \omega_0 \) which has changed phase by \( 180^\circ \). One quadrature sample for the digital receiver is computed by low pass filtering \( a_1 \) and \( -a_3 \); the other by filtering \( a_2 \) and \( -a_4 \). We use the Z
transform implementation for a low pass filter discussed earlier in this report. The algorithm for filtering the input samples follows.

\[
\begin{align*}
A(IDX) &= \text{input samples} \\
Y &= \text{quadrature component sample} \\
X &= \text{quadrature component sample} \\
\tau &= \text{low pass filter time constant} \\
CON &= h/\tau \\
QB &= -1.0 \\
Y &= 0.0 \\
X &= 0.0 \\
\text{DO 930 IDX } &= 1, 3, 2 \\
QB &= QB \\
y &= Y - \text{CON} \times Y + QB \times \text{CON} \times A(IDX) \\
930 \quad x &= X - \text{CON} \times X + QB \times \text{CON} \times A(IDX+1)
\end{align*}
\]

3.1.4 **Generation of PSK Modulated Noisy Carrier**

A data stream, for simulation purposes, was generated by a pseudo-random generator with variable length sequences. The data (NRZ or split-phase) was phase modulated on a carrier, and wideband noise with a known noise density was added to the carrier. For quadrature component sampling, the noisy carrier was then multiplied by an estimate of the carrier frequency and by a 90° phase shifted version. Both products were then low-pass filtered to eliminate the double-frequency terms (Figure 50a). The outputs of the low pass filters are the quadrature components of the noisy PSK carrier. The quadrature components and a third channel, which contains the data stream and the hard-line bit-synchronization pulses, are then A/D converted. For direct IF sampling, the noisy carrier was bandpass filtered and processed by the ADCOM IF Sampler (Figure 50b). The outputs of the IF Sampler are the quadrature samples of the noisy PSK carrier. The quadrature samples and a third channel, which contains the data stream and the hard-line bit synchronization pulses, are then A/D converted. As in the unmodulated...
carrier generation in Section 2.7, PSK carrier data was also generated by purely digital means to provide a check case for the bit error rates determined from the analog generated data.

By using digitally generated data, effects due to electronic instabilities in the A/D converter, noise generators, and the carrier oscillator are eliminated. The effects of different truncation levels were modeled in the CDC 6600 by internally reducing the number of bits/word to give a specified truncation error before entering the tracking loop. Figure 55 is a block diagram of the digital data generation subroutine which was used for generating the noisy PSK carrier.

3.1.5 Study of Carrier Synchronization

In order to demodulate PSK, it is necessary to estimate the phase and frequency of the subcarrier with as little error as possible. If the information process contains a residual component of sufficient strength at the subcarrier frequency, this component may be considered as a carrier and tracked as previously discussed. Several methods have been proposed for generating a reference subcarrier from the received signal when the residual subcarrier component is not available.

We will first consider the various analog carrier tracking techniques and then derive their digital equivalents. The squaring loop and the Costas loop which are common analog carrier tracking devices are shown in Figures 56 and 57. The analog Costas loop and squaring loop, although somewhat different in structure, are mathematically equivalent assuming that the effect of the presquaring bandpass filter is ignored. Neither method requires bit synchronization, but both are subject to 180 degree phase ambiguities. Since the two techniques are ideally mathematically equivalent, the digital Costas and squaring loop are identical. The digital Costas loop is developed later in this section.
DIGITAL SIGNAL GENERATION

DATA SOURCE

SIGNAL QUADRATURE COMPONENT GENERATION

NOISE GENERATOR

LOW-PASS FILTER

+ +

TRUNCATION

QUADRATURE COMPONENT OF NOISY CARRIER

+, +

DATA AND HARD-LINE BIT SYNC

QUADRATURE COMPONENT OF NOISY CARRIER
THE SQUARING LOOP FOR PHASE - TRACKING A BINARY - MODULATED SUPPRESSED - CARRIER INPUT

THE COSTAS LOOP FOR PHASE - TRACKING A BINARY PSK CARRIER

Figure 56

Figure 57
A third technique which has received considerable attention in the literature is the delayed decision feedback or the decision-directed carrier tracking. This method estimates the data bit and feeds this value back to be multiplied by a delayed version of the carrier (Figure 58). For this technique good bit-synchronization is required for proper performance. Unfortunately, most bit synchronizers also require good carrier synchronization to work. Thus a circular problem results and careful attention must be given to the interrelationship between bit and carrier tracking.
The digital decision-directed carrier tracking loop is more complex than the Costas loop in that a shift register must be provided to obtain a one bit-period time delay. A decision-directed carrier tracking loop also tracks with phase offset equal to $\omega_0 T$ radians where $\omega_0$ is the IF frequency and $T$ is the bit period. This requires that the bit period and carrier frequency be accurately known or that the modulation be synchronized to the carrier phase. Probability of error with our digital Costas loop and Decision Feedback loop is shown in Figure 59. Note that the slight performance enhancement obtained by the digital decision-directed carrier tracking loop does not appear sufficient to outweigh the previously discussed disadvantages. Also the digital Costas loop gives tracking performance which is comparable to the other techniques, but yet is considerably simpler to instrument. Using the Costas loop also separates the bit and carrier synchronization algorithms which simplifies acquisition. Thus, the Costas loop appears to be the best analog technique for digital implementation.

Synthesis of Carrier Synchronization Algorithm

The differential equations describing the operation of the Costas loop in terms of the quadrature component samples are derived in this section. The set of digital algorithms for carrier synchronization consists of a numerical solution of these equations.

The input to the Costas loop, $E(t) = x(t)\sin \omega_0 t + y(t)\cos \omega_0 t$, was defined in Section 3.1.1 in terms of the quadrature components. The lowpass filters in the Costas loop are assumed to have the following effects at points (1) and (2) in Figure 57:
DIFFERENTIALLY ENCODED DATA
HARD LINE BIT SYNC
NOISE BANDWIDTH = .05 Hz

BIT ERROR PROBABILITY

\[ \frac{E}{No} \text{ dB} \]

Figure 59
(1) The low frequency components of the signal and noise are passed without distortion;

(2) All double frequency components are completely rejected.

Let the output of the VCO be \( \cos(\omega_o t + \hat{\theta}(t)) \) where \( \omega_o/2\pi \) is the free running frequency of the VCO in Hz, \( \hat{\theta}(t) \) is the Costas loop estimate of the unknown phase \( \theta(t) \). Using these assumptions the voltages at 1 and 2 can be expressed in terms of the quadrature components as

\[
V_1(t) = -\frac{1}{2} x \sin \hat{\theta}(t) + \frac{1}{2} y \cos \hat{\theta}(t) \quad (119)
\]

\[
V_2(t) = \frac{1}{2} x \cos \hat{\theta}(t) + \frac{1}{2} y \sin \hat{\theta}(t) \quad (120)
\]

and the voltage \( e(t) \) is

\[
e(t) = \frac{1}{8} (y^2 - x^2) \sin 2\hat{\theta}(t) + xy \cos 2\hat{\theta}(t) \quad (121)
\]

The differential equation describing the loop is

\[
\frac{d\hat{\theta}(t)}{dt} = K F(p) e(t) \quad (122)
\]

where

\[
p = \frac{d}{dt} \text{ is the differential operator},
\]

\[
K = \text{multiplying constant for the VCO}, \text{ and}
\]

\[
F(p) = \text{transfer function for the loop filter}.
\]

Because the Costas loop must extract a reference in the presence of frequency detuning, the loop filter \( F(s) \) is chosen for a second-order active loop.

\[
F(s) = \frac{s+a}{s} \quad (123)
\]

The Costas loop characteristics are defined in terms of the damping factor \( \zeta \), the undamped natural frequency \( \omega_n \), and the noise equivalent bandwidth \( B_N \) of the linearized loop. These parameters can be defined in terms of various loop and signal constants.
A block diagram of the linearized Costas loop is given in Figure 60, where the input to the Costas loop is assumed to be signal only.

\[
\begin{align*}
\frac{\dot{\theta}(s)}{\theta(s)} &= \frac{\frac{A^2 K}{4}s + \frac{A^2 Ka}{4}}{s^2 + \frac{A^2 K}{4}s + \frac{A^2 Ka}{4}} \\
&= \frac{\frac{85\omega_n}{A^2}s + \omega_n^2}{s^2 + 2\zeta \omega_n s + \omega_n^2}
\end{align*}
\]

The transfer function can be written in the standard second order form.

\[
\frac{\ddot{\theta}(s)}{\dot{\theta}(s)} = \frac{2\zeta \omega_n s + \omega_n^2}{s^2 + 2\zeta \omega_n s + \omega_n^2}
\]

where \( K = \frac{85\omega_n}{A^2} \)

\( a = \frac{\omega_n}{2\zeta} \)

The damping factor \( \zeta \) is usually defined as \( \frac{1}{2} \) in Costas loops and that value will be used in the digital Costas loop. The undamped natural frequency \( \omega_n \) is defined in terms of the damping factor and the loop noise bandwidth \( B_n \).

\[
\omega_n = \frac{8\zeta B_n}{4\zeta^2 + 1}
\]
The loop noise bandwidth $B_N$ is defined from linear phase-locked loop theory as

$$B_N = \frac{1}{2} \int_{-\infty}^{\infty} \frac{\hat{\theta}(f)}{\hat{\theta}(f)} \, df$$  \hspace{1cm} (129)$$

where $\frac{\hat{\theta}(f)}{\hat{\theta}(f)}$ is defined by Equation (124) with $s = j2\pi f$.

If the constant amplitude factor $A$ and the loop noise bandwidth $B_N$ are known, the Costas loop gain $K$ and loop filter constant $a$ can be determined. It is convenient to describe the Costas loop in terms of its linear parameters even when it is operating in the nonlinear range. The Costas loop can be completely described in the time domain by substituting Equations (121) and (123) into (122)

$$\frac{d^2\hat{\theta}(t)}{dt^2} = Ka \left[ \frac{1}{8} (y^2-x^2) \sin 2\hat{\theta}(t) + \frac{1}{4} xy \cos 2\hat{\theta}(t) \right] + K \frac{d}{dt} \left[ \frac{1}{8} (y^2-x^2) \sin 2\hat{\theta}(t) + \frac{1}{4} xy \cos 2\hat{\theta}(t) \right]$$  \hspace{1cm} (130)$$

The real time solution of Equation (130), where $x$ and $y$ are the signal quadrature components, is the real time estimate of the carrier phase which is mathematically equivalent to that of an ideal Costas loop.

To obtain the real time solution for $\hat{\theta}(t)$, Equation (130) can be represented by two first order simultaneous differential equations. Let $X_1 = \hat{\theta}$ and $X_2$ be a dummy variable, Equations (131) and (132) are equivalent to Equation (130).

$$\frac{d}{dt} X_1 = X_2 + K \left[ \frac{1}{8} (y^2-x^2) \sin 2X_1 + \frac{1}{4} xy \cos 2X_1 \right]$$  \hspace{1cm} (131)$$

$$\frac{d}{dt} X_2 = Ka \left[ \frac{1}{8} (y^2-x^2) \sin 2X_1 + \frac{1}{4} xy \cos 2X_1 \right]$$  \hspace{1cm} (132)$$

The real time numerical solution of Equations (131) and (132) is the digital equivalent of the analog Costas loop.
A very simple numerical integration routine must be used to solve Equations (131) and (132) because all signal processing in the digital receiver is done in real time. Consequently, very complex integration routines that involve higher order derivatives like Taylor's algorithm \(^1\) are not applicable. The Runge-Kutta method of order \(^1\) is often used in the numerical integration of differential equations. This method requires four derivative evaluations per step. For real time calculations the computation time necessary for four derivative evaluations makes the Runge-Kutta method impractical. To satisfy the requirement for the least number of computations per step, Euler's method \(^1\) was chosen. This method requires a small step size or high sampling rate, but uses only one derivative evaluation per step. In general, Euler's method for the solution of the differential equation

\[
\frac{db}{dt} = f(a,b)
\]

is

\[
b = b_{n-1} + hf(a_{n-1}, b_{n-1})
\]  

where \(h\) is the step size.

The solution of the Costas loop equation by Euler's method where \(H\) is the sample interval is given in Equations (135) and (136).

\[
X_{1n} = X_{1n-1} + H \left[ X_{2n-1} + \frac{1}{8} (Y_{n-1}^2 - X_{n-1}^2) \sin 2X_{1n-1} + \frac{1}{4} X_{n-1}Y_{n-1} \cos 2X_{1n-1} \right]
\]  

(135)

\[
X_{2n} = X_{2n-1} + H \left[ K \frac{1}{8} Y_{n-1}^2 - X_{n-1}^2 \right] \sin 2X_{1n-1} + \frac{1}{4} X_{n-1}Y_{n-1} \cos 2X_{1n-1} \right]
\]  

(136)
Equations (137) and (138) are difference equations for the Costas loop phase estimate and can be solved in real time on a digital computer. The computer algorithm for the Costas loop follows:

\[ H = \text{Sample Interval} \]
\[ D_k = H \cdot k \]
\[ C_k = H \cdot k \cdot a \]
\[ X_1 \text{ IS COSTAS PHASE ESTIMATE} \]
\[ X_2 \text{ IS A DUMMY VARIABLE} \]
\[ ADC(1) \text{ AND ADC(2) ARE THE DIGITIZED QUADRATURE COMPONENTS} \]

\[ X = ADC(1) \]
\[ Y = ADC(2) \]
\[ \text{TEMP} = (Y \cdot Y - X \cdot X) \cdot \sin(2 \cdot X_1) / 8. \]
\[ + X \cdot Y \cdot \cos(2 \cdot X_1) / 4. \]
\[ X_1 = X_1 + H \cdot X_2 + \text{TEMP} \cdot D_k \]
\[ X_2 = X_2 + \text{TEMP} \cdot C_k \]

The real time phase estimate from the digital Costas loop is used to establish a coherent reference for the cross-correlation or matched filtered detection process. The receiver cross-correlation operation is performed by a multiplier-integrator combination that multiplies the signal with the coherent reference and integrates over a bit period. The correlator output \( C \) is given by Equation (137).

\[ (n+1)T \]
\[ C = \int_{nT}^{(n+1)T} E(t) \sin(\omega_0 t + \hat{\theta}(t)) \, dt \]  

\[ (137) \]

where \( T \) is a bit period. Since the integration in the correlator averages out the double frequency terms, Equation (137) can be expressed in terms of the quadrature components as Equation (138).

\[ (n+1)T \]
\[ C = \int_{nT}^{(n+1)T} \frac{1}{2} (x \cos \hat{\theta}(t) + y \sin \hat{\theta}(t)) \, dt \]  

\[ (138) \]

The digital correlator output is computed by performing the integration using the quadrature components and the phase estimate from the digital Costas loop. A bit decision is made by determining \( \text{sgn}[C] \). The timing
command to make a decision on the correlator must be supplied from the bit synchronization algorithm. The digital algorithm for the correlator, performed using Euler's method is Equation (139).

\[ C_n = C_{n-1} + H \left( \frac{1}{2} (X_{n-1} \cos \frac{Y_{n-1}}{n-1} + Y_{n-1} \sin \frac{X_{n-1}}{n-1}) \right) \]  

(139)

3.1.6 Automatic Gain Control

Earlier in this report we discussed two AGC techniques for use with a phase-locked loop. In the first approach we placed a bandpass limiter at the input of the phase-locked loop. In the second method we utilized a closed loop AGC preceding the phase-locked loop. We chose the bandpass limiter as the preferred method for use with a phase-locked loop because of its simplicity and the small computation time required. In addition, this technique will correct amplitude variations instantaneously. In this section we will apply the limiter to our digital receiver to provide AGC for the Costas loop.

We incorporated the bandpass limiter into the Costas loop as we did in the phase-locked loop in Section 2.5.1. With a bandpass limiter the voltage \( e(t) \) (Figure 57) is Equation (140).

\[ e(t) = \frac{1}{8} \frac{y^2 - x^2}{x^2 + y^2} \sin 2\dot{\theta}(t) + \frac{1}{4} \frac{xy}{x^2 + y^2} \cos 2\dot{\theta}(t) \]  

(140)

The Costas loop equation with a bandpass limiter is Equation (141).

\[ \frac{d^2 \dot{\theta}(t)}{dt^2} = K_a \frac{1}{8} \frac{y^2 - x^2}{x^2 + y^2} \sin 2\theta(t) + \frac{1}{4} \frac{xy}{x^2 + y^2} \cos 2\theta(t) \] 

\[ + K_b \frac{d}{dt} \left[ \frac{1}{8} \frac{y^2 - x^2}{x^2 + y^2} \sin 2\theta(t) + \frac{1}{4} \frac{xy}{x^2 + y^2} \cos 2\theta(t) \right] \]  

(141)
To obtain the real time solution for $\dot{\Theta}(t)$, Equation (141) can be represented by two first order differential equations. Let $X_1 = \dot{\Theta}$ and $X_2$ be a dummy variable. Equations (142) and (143) are equivalent to Equation (141).

$$\frac{d}{dt} X_1 = X_2 + K \frac{1}{8} \frac{y^2 - x^2}{x^2 + y^2} \sin 2X_1 + \frac{1}{4} xy \cos 2X_1$$  \hspace{1cm} \text{(142)}$$

$$\frac{d}{dt} X_2 = Ka \frac{1}{8} \frac{y^2 - x^2}{x^2 + y^2} \sin 2X_1 + \frac{1}{4} xy \cos 2X_1$$  \hspace{1cm} \text{(143)}$$

The real time numerical solution of Equations (142) and (143) is the digital equivalent of the analog Costas loop with a bandpass limiter. The numerical solution of Equations (142) and (143) using Euler's method where $H$ is the step size, is given in Equations (144) and (145).

$$X_2^*_n = X_1^{n-1} + H \left[X_2^{n-1} + K \frac{1}{8} \frac{y_{n-1}^2 - x_{n-1}^2}{x_{n-1}^2 + y_{n-1}^2} \sin 2X_1^{n-1} + \frac{1}{4} \frac{x_{n-1}y_{n-1}}{x_{n-1}^2 + y_{n-1}^2} \cos 2X_1^{n-1} \right]$$  \hspace{1cm} \text{(144)}$$

$$X_2^*_n = X_2^{n-1} + H \left[K \frac{1}{8} \frac{y_{n-1}^2 - x_{n-1}^2}{x_{n-1}^2 + y_{n-1}^2} \sin 2X_1^{n-1} + \frac{1}{4} \frac{x_{n-1}y_{n-1}}{x_{n-1}^2 + y_{n-1}^2} \cos 2X_1^{n-1} \right]$$  \hspace{1cm} \text{(145)}$$

The computer algorithm for the Costas loop with a bandpass limiter follows.

$$H = \text{Sample Interval}$$

$$DK = H*K$$

$$CK = H*K*a$$

$X_1$ IS COSTAS PHASE ESTIMATE

$X_2$ IS A DUMMY VARIABLE

$\text{ADC(1)}$ AND $\text{ADC(2)}$ ARE THE DIGITIZED QUADRATURE COMPONENTS

$$X = \text{ADC(1)}$$

$$Y = \text{ADC(2)}$$

$$\text{TEMP} = (Y**2 - X**2) * \sin (2.*X1)/(X**2 + Y**2)/8.$$  \hspace{1cm} + X*Y*cos(2.*X1)/(X**2 + Y**2)/4.

$$X1 = X1 + H**2 + \text{TEMP} * DK$$

$$X2 = X2 + \text{TEMP} * CK$$

A graph showing the output phase variance as a function of the input noise spectral density for the Costas loop with, and without, a limiter is shown in Figure 61. This curve shows that the phase variance is
COSTAS LOOP BANDWIDTH - 0.75 Hz
BIT RATE - 10 BITS/SECOND
SAMPLE PERIOD - 0.005 SECONDS
SIMULATION LENGTH - 10,000 BITS

COSTAS LOOP WITHOUT BANDPASS LIMITER

COSTAS LOOP WITH BANDPASS LIMITER

PHASE ERROR VARIANCE - RADIANS²

Figure 61
nearly identical for both cases. Addition of the bandpass limiter does not
degrad Costas loop operation, and allows the bandwidth of the Costas loop
to be set independently of the input signal amplitude.

3.1.7 Study of Bit Synchronization

One of the objectives of this study is to determine an "optimum" bit
synchronizer for use with the MDAC digital receiver. This bit synchronizer
must also be "easily realizable" on a digital computer and must adapt to
relatively large bit rate variations.

To develop an "optimum" bit synchronizer, a model of the input data
stream and a criterion of optimality is required. The model of the input data
stream must specify the symbol waveforms, the symbol occurrence statistics and
the statistics of the timing jitter. Previously, using nonlinear filter theory,
we \textsuperscript{11, 12, 13}, synthesized a bit synchronizer and detector which was optimum in
the sense of minimum bit error probability for the case of a constant known
bit rate, but an unknown phase. Other authors \textsuperscript{14, 15, 16}, have considered slight-
ly less general situations using Bayesian or maximum likelihood estimation and
have obtained somewhat similar results. Although none of these authors direct-
ly consider the case of varying or unknown bit rate, this case can also be
treated by a conceptually straightforward generalization of the previous
results.

The second constraint on our bit synchronizer is that it be "easily
realizable" on the digital computer. In a certain sense, any system which can be
mathematically defined can be implemented essentially exactly using a digital
computer and at least approximately using analog hardware. The optimum bit
synchronizer defined in reference \textsuperscript{11} is certainly physically realizable in the
classical mathematical sense in that it does not require operations on data
from future time. However, from a practical hardware implementation viewpoint it is not realizable for normal bit rates. Even assuming a known bit rate of $1/T$ bits/seconds, the implementation of the optimum bit synchronizer would require a bank of $4M$ correlators (i.e., matched filters) to achieve a minimum timing error of $T/M$ seconds. If the bit rate is also unknown, the number of correlators required is increased by another factor which is related to the uncertainty in bit rate.

A normal analog bit synchronizer at high signal-to-noise ratios will operate with a timing error of less than 5% which would require that $M$ be greater than 20 to get equivalent performance. Thus with no uncertainty in bit rate, at least 80 correlators are required for the bit sync alone. Section 4 discusses this approach in detail and presents experimental performance data.

Suppose we now attempt to synthesize the "optimum realizable" bit synchronizer directly. What really do we mean by "realizable"? We propose that by "realizable", we really mean practically implementable at reasonable bit rates. Since this definition of "realizable" is not a mathematically precise definition, the "optimum realizable" solution cannot be determined directly from a mathematical argument. However, there is an analytical approach which leads to physically meaningful results and this approach will now be outlined.

If we look at the various bit synchronizers that have been implemented, we find that they all resemble phase-locked loops in the sense that they all make an estimate of the phase difference or error between the incoming bit stream and some internally generated reference, filter this error estimate, and then update the internal reference based on the filtered error value. In fact, the optimum bit synchronizers discussed earlier can also be thought of in this way.
if the idea of a phase detector and filter are each generalized somewhat. Therefore, we have investigated two approaches to bit synchronization which utilize a phase detector and loop filter combination. The first type utilizes a nonlinearity to generate a frequency component at the bit rate and tracks this component with a phase-locked loop. The second approach utilizes an early and late gate to generate an error signal which is used in a feedback loop to center the gates on the transition. In the following sections we will discuss two implementations of each type of bit synchronizer and compare them by determining timing jitter as a function of signal-to-noise ratio and mean acquisition time as a function of frequency offset.

Nonlinear Bit Synchronization

We will investigate two types of bit synchronizers which fall into this category. The first technique, which was considered by Wintz and Luecke 14, consists of filtering, squaring, and bandpass filtering the bit stream and utilizing positive zero crossing of the resultant signal. In general for the case where the bit rate is unknown, the bandpass filter would be replaced by a phase-locked loop as shown in Figure 62.

![Square Law Bit Synchronizer Diagram](image)
The input low pass filter is used to obtain a maximum signal-to-noise ratio at the input to the square law detector. The square law detector generates a frequency component at the bit rate which is tracked by the phase-locked loop.

In order to analyze this technique we must first determine the amplitude of the harmonic generated at the bit rate. This amplitude can be determined by computing a Fourier series of the square law detector output. For the specified square pulse shape with either NRZ or Manchester coded input data, the amplitude of this component is zero. However, this does not negate the use of this approach, since if the input signal is first differentiated and then squared, a frequency component at the bit rate will be generated. The performance of this approach is then determined by computing the amplitude of the fundamental component using Fourier analysis. The ratio between the power at the bit rate and the total power in the bit train is a measure of power penalty with this approach. This technique also has the disadvantage that the amplitude of the frequency component at the bit rate is a function of the input bit sequence. A string of ones or zeros will reduce this amplitude to zero for short intervals of time. This effect reduces the tracking capability of the loop and increases the output phase variance. To illustrate the operation of this technique, we assume for the moment that a one is represented by a positive half sinusoid and a zero by a negative half sinusoid. If we assume that the low pass filter does not alter the shape of the input pulse, the output of the square law detector is

\[ V(t) = A^2 \sin^2 \omega t = \frac{1}{2} A^2 (1 - \cos 2 \omega t) \]  

\[ A = \text{peak bit amplitude} \]

\[ \omega = \frac{\pi}{T} \]

\[ T = \text{bit period} \]
The phase-locked loop following the square law detector tracks the frequency component at the bit rate \(2^{\frac{w}{n}}\) (amplitude \(A^2/2\)) to obtain the bit timing reference.

The second approach we investigated generates a frequency component at the bit rate by multiplying the input signal by itself delayed by one half of the bit period. A block diagram of this method is given in Figure 63. In contrast to the previous technique, this approach will generate a harmonic at the bit rate for square input pulses with either the NRZ or Manchester format. If the input bits have amplitude \(A\) and ones and zeros are equiprobable, it can be shown that the average autocorrelation function has the form shown in Figure 64.
The average time waveform associated with the periodic component of this auto-
correlation function is

\[ V(t) = \frac{2A^2}{\pi} \sum_{n=1,3,5,...} \frac{1}{n} \sin \left( \frac{2n\pi t}{T} \right) \]  

The phase-locked loop following the multiplier will track the frequency com-
ponent at the bit rate to obtain the clock signal. For NRZ data the amplitude
of this component is time varying because it depends on the bit sequence. For
extremely low loop bandwidths this effect will be negligible since the vari-
ance in amplitude will be averaged. As with the square law implementation,
for wider loop bandwidths the tracking effectiveness of the loop will be de-
creased and the output phase variance will be increased.

**Gated Bit Synchronization**

The final type of bit synchronizer which we investigated in this section
is a gated phase-locked loop. This technique uses two overlapped gates around
the transition as shown in Figure 65.

The first gated technique that we considered was developed by Layland\(^{17}\)
and is shown in Figure 66. In this technique the signal in the early and late
gates is integrated, squared, and subtracted to determine the error signal,
which is then filtered and used to drive the VCO.

We also investigated a second gated approach, which was suggested by
Simon\(^{18}\), that determines the error signal by subtracting the absolute values
of the early and late gates outputs. Both of these approaches can be used
with either NRZ or Manchester coded input data. Since the gate width must be
reduced to one half of the bit period for the latter type data, the possibility
of locking-up on the wrong transition exists, and the linear region of the
error curve will be reduced.
Figure 65

GATE LOCATION

Figure 66

GATED BIT SYNCHRONIZER
Comparison of Techniques

We first determined the rms bit jitter as a function of the signal-to-noise ratio for the square law and absolute value gated bit synchronizers and the delay and multiply nonlinear synchronizer. The square law nonlinear implementation was omitted because its operation is essentially the same as the delay and multiply approach. The resulting data, which is shown in Figure 67, was obtained using 20 samples per bit and an equivalent loop time constant of 10 bit periods. This data shows that there is very little difference between any of the three approaches. The absolute value method has lower rms tracking error over the total range of signal-to-noise ratios. For high signal-to-noise ratios the delay and multiply method ranks second, while at the low signal-to-noise ratios the square law implementation is second best.

We next determined the acquisition time as a function of the percent uncertainty in bit rate. The main problem in obtaining this data is defining when acquisition has occurred. In our digital simulation, we utilized 80 samples per bit and defined acquisition as having occurred when the absolute error remained less than 2 samples (1/40 of the bit period) for a total of 10 bits (i.e., one loop time constant). We then designed each of the bit synchronization loops to operate with a nominal bit period of 80 samples and a time constant of 10 bit periods. Figure 68 shows a graph of the pull-in time in bit periods as a function of the bit period increase in percent. These results were obtained using a noise free random bit sequence of equally probable ones and zeros. This data indicates that the square law gated implementation gives the best acquisition time. The other two approaches are approximately equivalent.
COMPARISON OF BIT SYNCHRONIZERS
LOOP TIME CONSTANT = 10 BIT PERIODS

Figure 67
In determining the best approach to use for the MDAC Digital PSK receiver, we rated pull-in time as an important performance criterion. The rms bit jitter for each of the techniques can be equalized by increasing the signal-to-noise ratio, but the acquisition times which were shown in Figure 68 were taken for the noise free case and thus cannot be further improved. With this in mind, we selected the square law gated implementation since it gives the optimum pull-in time and is only slightly worse than the absolute value implementation with respect to bit jitter.

Synthesis of Bit Synchronization Algorithm

In this section we derive the algorithms for the bit synchronization loop. We first derive an algorithm for NRZ data, and then, using a similar approach, derive an algorithm for split-phase data.

Synthesis of Bit Synchronization Algorithm - NRZ Data

A functional diagram of the MDAC Digital PSK bit synchronization loop, as modeled after the telemetry bit synchronization loop developed by JPL, is shown in Figure 69. The bit synchronization loop operates by sampling the early and late integrals of the input NRZ data stream; squaring the two integrals; differencing and filtering the two squared values to obtain a phase-error correction; and then generating the early gate, late gate, and bit timing commands.

The early integral is the integration of the input data stream from one quarter of a bit period before the local estimate of the data transition time, $t_1$, to three quarters of a bit period after the estimated data transition time. The late integral is the integration of the data stream from three quarters of a bit period before the estimated data transition time to one quarter of a bit period after the estimated data transition time. The sign
of the two integrals is removed by squaring so that the bit sync loop error signal is independent of the polarity of the data transitions. $E_i$, the loop phase error signal, is generated by differencing the squares of the two integrals. The value of this difference will be zero (for the noise free case) when no data transitions have occurred during the early and late integrals, or when a data transition occurred at the same time as the local estimate of the transition time. When the difference of the two squares is not zero, there is a phase error in the local estimate of the transition time. The sign of the difference indicates the direction to shift the phase of the local estimate of the transition time to correct the phase error.
The magnitude of the difference is proportional to the amount of phase shift necessary to correct the phase error.

For example consider the case where the local estimate of \( t_1 \) is off by one quarter of a bit period. Let \( t_1 \) denote the actual transition time and \( T \) denote the bit period. Then \( t_1 + \frac{1}{4} = t_1 \). Define \( v(t) = +1 \) for \( t < t_1 \) and \( v(t) = -1 \) for \( t > t_1 \). Thus the late integral is \( +T \) and the early integral is zero. The error signal, \( E_i \), is then \( +2^2 \). If \( v(t) = -1 \) for \( t < t_1 \) and \( v(t) = 1 \) for \( t > t_1 \), the same error signal is obtained; thus the error signal is data independent. Now let the local estimate of \( t_1 \) be off by one quarter of a bit period in the opposite direction so that \( t_1 - \frac{1}{4} = t_1 \). Let \( v(t) = +1 \) for \( t \leq t_1 \) and \( v(t) = -1 \) for \( t > t_1 \). The late integral is now zero and the early integral is \(-T \) resulting in \( E_i = -2^2 \). Changing the sign on the transition as before does not change the error signal. When \( t_1 = t_1 \) the difference of squares error signal is zero regardless of the sign of the transition. Similarly the error signal is zero if \( v(t) \) has no transition between \( t_1 - 3/4 \) and \( t_1 + 3/4 \).

The error signal, \( E_i \), computed in this manner is linear for \( t_1 - \frac{1}{4} \leq t_1 \leq t_1 + \frac{1}{4} \) as shown in Figure 70. Outside this region the sign of the error signal corresponds to the direction to shift the estimate of the transition time to correct the phase error in a minimum number of steps.

Had the integration windows been chosen so that the early and late integrals were computed over an interval of less than one bit period each, the error signal would have a smaller linear range.

For computer implementation the phase error signal, \( E_i \), is scaled to an equivalent number of discrete phase steps. This number is added to the number of samples in the basic bit period which must be applied apriori to the computer program. The phase shifter accepts this output consisting of
the number of samples in the basic bit period combined with the number of samples of phase correction necessary to synchronize the local data clock with the actual data. This value is counted down to zero and the phase shifter is again ready to accept new timing information. When no phase correction is indicated the phase shifter merely counts down the number of samples in the basic bit period and outputs the timing commands for the early gate, late gate, and data integrals. We considered two different procedures when a phase correction is necessary. In the first case we reduce computation time by not storing any samples of the input signal. To operate without storing input samples, a computation of the error signal must be omitted each time the gates are moved while new samples are read into the gates. The phase shifter outputs only the data timing commands, and no error signal is computed during that data interval. This approach may degrade tracking an unknown bit rate since for a large bit rate offset the gate center is constantly being slewed. If the offset is large enough, the gate center is moved on each sample. With this implementation the gate can only move on every other sample since an error calculation is skipped after a move occurs.
In the second approach, this problem is eliminated by a more complex technique in which input samples are stored and an error signal is computed on each bit. All the input samples are stored from the bit period prior to an error signal calculation. When a phase correction is necessary, the phase shifter uses the stored samples and the new value of $\hat{t}_i$ to update the gate values. The phase shifter outputs the data timing commands and gate timing commands for the next error signal. We developed computer algorithms for both approaches and have compared their performance in this section. We will first develop the algorithm which does not store samples. The relative positions of timing marks for the early, late, and data integrations are shown in Figure 71 for perfect synchronization.

\[ E_{i+1} = (A + B)^2 - (B + C)^2 \]  
\[ E_{i+2} = (C + D)^2 - (D + E)^2 \]

Figure 71 shows that the early and late integrals overlap by one half of a bit period.

In order to reduce computations the overlapping integration can be removed by generating bit timing commands so that there are three integration windows each one half bit period long as shown in Figure 72.
The early and late integrals are then determined by Equations (150) and (151) where the overlapping integrations have been removed.

\[
\hat{t}_{i+1/4} + \frac{3}{4} = \hat{t}_{i+3/4} + \int_{\hat{t}_{i+1/4}}^{\hat{t}_{i+3/4}} v(t) \, dt
\]

\[
\hat{t}_{i-1/4} + \frac{1}{4} = \hat{t}_{i-3/4} + \int_{\hat{t}_{i-1/4}}^{\hat{t}_{i-3/4}} v(t) \, dt
\]

(150)
\[\int_{t_i - 3/4}^{t_i + 1/4} v(t) \, dt = \int_{t_i - 3/4}^{t_i - 1/4} v(t) \, dt + \int_{t_i - 1/4}^{t_i + 1/4} v(t) \, dt \quad (151)\]

Let \(A_i\), \(B_i\), and \(C_i\) be defined by Equations (151), (152), and (153).

\[
A_i = \int_{t_i - 3/4}^{t_i - 1/4} v(t) \, dt \quad (152)
\]

\[
B_i = \int_{t_i - 1/4}^{t_i + 1/4} v(t) \, dt \quad (153)
\]

\[
C_i = \int_{t_i + 1/4}^{t_i + 3/4} v(t) \, dt \quad (154)
\]

The A integral on the \(i + 1\) data bit is identical to the C integral on the \(i\)th data bit. Therefore additional computation time may be saved by updating the A integral as shown in Equation (155).

\[
A_i + 1 = C_i \quad (155)
\]

The video input, \(v(t)\), for the bit synchronization loop is generated from quadrature components, \(x\) and \(y\), and the Costas loop phase output,

\[
v(t) = x \cos \hat{\theta} + y \sin \hat{\theta} \quad (156)
\]

Using this expression for \(v(t)\) the \(A_i\), \(B_i\), and \(C_i\) integrals may be evaluated by rectangular integration. The \(A_i\), \(B_i\) and \(C_i\) integrals can be defined in terms of notation previously employed.
\[ A_n = A_{n-1} + H[x_{n-1} \cos X_{n-1} + y_{n-1} \cos X_{n-1}] \]
\[ B_n = B_{n-1} + H[x_{n-1} \cos X_{n-1} + y_{n-1} \cos X_{n-1}] \] (157)
\[ C_n = C_{n-1} + H[x_{n-1} \cos X_{n-1} + y_{n-1} \cos X_{n-1}] \]

It is required that the bit synchronization loop be able to acquire and track a ramp input. Thus, the bit synchronization loop must use a second order filter. Choose the loop filter of Figure 72 as \( F(s) = \frac{s + a}{s} = 1 + \frac{s}{a} \)

Then the feedback loop of Figure 72 can be represented as shown in Figure 73, where the gain \( G \) is chosen to scale the error signal to an integer number of discrete phase steps. \( G \) is then \( \frac{SR}{4T^2} \) where \( SR \) is the number of samples per bit.

![Diagram](attachment:bit_synchronization_feedback_loop.png)

**BIT SYNCHRONIZATION FEEDBACK LOOP**  
**Figure 73**

The loop transfer function of Figure 73 is given by Equation (158).

\[ H(s) = \frac{Gk(s + a)}{s^2 + Gks + GKA} \]  
(158)

Equation (158) is in the standard second order form given in Equation (159)

\[ H(s) = \frac{2\zeta_0 \omega_0 s + \omega_0^2}{s^2 + 2\zeta_0 \omega_0 s + \omega_0^2} \]  
(159)

The loop time constant, \( T_{CON} \), is approximately \( \frac{1}{\zeta \omega_0} \).
Equation (160) follows from Equation (158) and (159).

\[ k = \frac{2\zeta \omega_0}{G} \]  
\[ a = \frac{\omega_0}{Gk} \]  

Since \( E_i \) is zero one half the time due to no transitions in the data stream, the integrator gain \( k \) is doubled to compensate for the loss of error signal. Thus,

\[ k = \frac{4\zeta \omega_0}{G} \]  

The digital bit synchronization algorithm can be derived as follows. Let \( E_i \) be the input to the loop filter. Since the \( \frac{a}{s} \) term in the loop filter and the \( \frac{k}{s} \) term in the VCO represent integrations in the time domain, the algorithm for the bit synchronization loop, using rectangular integration, can be shown to be given by Equation (162).

\[ E = ERR + EI * T \]

\[ ERR = E \]

\[ ERRR = k*a*E + k*EI \]

\[ THAT = THAT + ERRR * T \]

where \( EI = E_i \), \( k \) and \( a \) are defined as before, \( T \) is the bit period, and \( THAT \) is the final filtered output.

The input for the phase shifter is formed by first converting \( THAT \) to the nearest whole number of samples \( NTHAT \), and computing the difference \( IR = NTHAT_n - NTHAT_{n-1} \). This integer number is the estimated number of samples of phase shift required to align the transition times. It is the
change in NTHAT from the n-1 error signal to the nth error signal that represents the phase correction, because the phase shifter uses its present estimate of the transition time as a reference and can move only in integer steps away from this position to correct its estimate of the transition time. The phase shifter adds or subtracts this difference, IR, to the number of samples in the basic bit period and counts down this number to correct the local estimate of the transition time. Therefore, if there is a ramp input caused by incorrect knowledge of the bit period, the phase shifter tracks the ramp by changing the basic bit period by IR samples.

To generate the timing commands, the phase shifter relies on a knowledge of the number of samples per bit and the integration intervals for the A₁, B₁, and C₁ integrals. Reference to Equations (152), (153), and (154) and Figure 71 shows that the integration window for the early and late integrals is 1.5 bit periods long. Each integral, A₁, B₁, and C₁, is over half of a bit period and thus contains .5 SR samples. The phase shifter has a counter, I, that begins at tᵢ - 3/4, counts 1.5 SR samples, and then is reset to zero. The A₁ integral is computed from tᵢ - 3/4 to tᵢ - 1/4 over the first .5 SR samples; the B₁ integral is computed from tᵢ - 1/4 to tᵢ + 1/4 while C₁ is computed from tᵢ + 1/4 to tᵢ + 3/4 over the last .5 SR samples.

E₁ is then computed, filtered, and a decision is made on IR to correct the phase. If IR = 0, A₁ + 1 is set equal to C₁ and the counter I is advanced so that B₁ + 1 and C₁ + 1 are computed. If it is necessary to correct the local estimate of the transition time, then a slightly different procedure is followed. E₁ is computed at tᵢ + 3/4 following calculation of C₁ as shown in Figure 71. The phase shifter corrects by increasing or decreasing the number of samples between tᵢ + 3/4 and tᵢ + 1. No error signal Eᵢ + 1
is calculated because it would have been necessary to save and order the samples received while the phase correction was being made to obtain the correct $A_{i+1}$ and $B_{i+1}$ integrals. Therefore the phase shifter waits until the $A_{i+2}$ integral is to be computed and again begins outputting timing commands.

The phase shifter has a second counter IDUM that generates the timing command for the correlator integral by counting the number of samples in the basic bit period starting at $t_i$. When SR samples are counted the bit decision is made and the correlator integrator reset. When a phase correction is necessary IDUM is advanced or retarded so that the correlation integrates over less than or more than a bit period as indicated by the error signal. In this manner the bit synchronization loop is able to adjust the timing commands of the data, early, and late integrals to correct the phase estimate of the local clock without storing input samples.

The sample storage approach for bit synchronization is identical to the approach just described except the action taken when a phase correction is necessary. In the second approach all the input samples from the bit period prior to an error signal calculation (from $t_i - 1/4$ to $t_i + 3/4$) are stored and ordered in the computer. After the error signal, $E_i$, is computed at $t_i + 3/4$, the phase shifter uses the new value of $t_i$ to update the gate integrals for calculating $E_{i+1}$. Depending on the sign and the magnitude of $E_i$, all or portions of $A_{i+1}$ and $B_{i+1}$ integrals are calculated from the stored samples. The counter, I, is advanced so that the next input sample goes to the proper gate A or B. The correlator timing signal IDUM is set to the proper value from the new value of $t_i$, and a bit decision is made as indicated by the corrected value of IDUM. IDUM is advanced or retarded
so that the correlator integrates over less than or more than a bit period as indicated by the error signal.

In this manner the bit synchronization loop is able to adjust the timing commands of the data, early, and late integrals to correct the phase estimate of the local clock. By using the stored input samples, the bit synchronizer updates the gate integrals after an error signal calculation, and, thus, an error signal is calculated on every bit.

A curve of bit error rate for the digital receiver with both bit synchronization techniques is shown in Figure 74. The bit error rate is slightly lower when using the bit synchronizer with sample storage. A curve of standard deviation of bit jitter is shown in Figure 75. Bit jitter is also lower when using the sample storage bit synchronization algorithm. We investigated tracking performance of the bit synchronization loops for cases having a large uncertainty in bit rate. We found that without sample storage, the bit synchronization loop time constant must be 5 bit periods or less for the loop to track a 5% uncertainty in bit rate. The bit synchronization loop with sample storage will track the 5% uncertainty in bit rate with a loop time constant of 10 bit periods or less, and will track a 10% uncertainty in bit rate with a loop time constant of 8 bit periods or less. Without sample storage, the bit synchronization loop will not track a 10% uncertainty in bit rate because of the one and a half bit delay after each correction of the bit timing reference before a new error signal can be computed. Sample storage allows the digital receiver to operate over a larger uncertainty in bit rate than was possible previously and significantly reduces bit tracking error (Figure 76).
BIT ERROR RATE FOR DIGITAL RECEIVER WITH AND WITHOUT SAMPLE STORAGE BIT SYNCHRONIZER

Figure 74

DIGITAL RECEIVER WITH SAMPLE STORAGE BIT SYNCHRONIZER

DIGITAL RECEIVER WITH NON SAMPLE STORAGE BIT SYNCHRONIZER

THEORETICAL PSK
Figure 75

**Standard Deviation of Bit Jitter**

- **Costas Loop Bandwidth**: 0.75 Hz
- **Bit Sync Loop Time Constant**: 100 bit periods
- **Bit Rate**: 10 bits/second
- **Sample Period**: 0.005 seconds
- **Simulation Length**: 10,000 bits

**E/N - dB**

**Standard Deviation of Bit Jitter**
DIGITAL RECEIVER STUDY AND IMPLEMENTATION

COSTAS LOOP BANDWIDTH: 0.0625 Hz
SAMPLE PERIOD: 0.005 SECONDS
ACTUAL BIT PERIOD: 0.095 SECONDS
EXPECTED BIT PERIOD: 0.100 SECONDS
SIMULATION LENGTH: 10,000 BITS

STANDARD DEVIATION OF BIT JITTER - 5% TIMING ERROR

BIT SYNC WITHOUT SAMPLE STORAGE
TIME CONSTANT = 6 BIT PERIODS

BIT SYNC WITH SAMPLE STORAGE
TIME CONSTANT = 10 BIT PERIODS

\[ \frac{E}{N_0} \text{ dB} \]

0 4 8 12 16 20 24 28 32

STANDARD DEVIATION OF BIT JITTER-FRACTION OF BIT PERIOD

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST

Figure 76
Synthesis of Bit Synchronization Algorithm Split-Phase Data

We modified our (NRZ) sample storage bit synchronization algorithm to operate with split-phase data by adding an "M-out-of-N" detector to determine correct phase. Split-phase data has a transition in the middle of each bit. The bit synchronizer must center the bit timing on this transition and not on a transition which may or may not occur at the end of the bit.

For split-phase data, we operate our bit synchronizer at twice the split-phase bit rate. Each half of the information bit is detected, and thus an error signal, $E_i$, is generated on each half bit. The phase shifter generates timing commands in exactly the same manner as for NRZ data for the early, late, and data integrals on each one half data bit. The phase shifter also controls a correlator integral over the entire information bit; and if necessary, corrects timing for this integral after each error signal is computed.

To center the bit timing correctly a counter is used to determine the number of mid-bit transitions in $N$ bit periods. If this number is above the threshold setting after $N$ bit periods, correct bit timing is assumed. When the number of transition is less than the threshold setting, the phase shifter changes its estimate of the mid-bit transition by 180° and initializes all counters.

With an "M-out-of-N" detector the probability of false acquisition and the probability of acquisition are defined in Equations (163) and (164).

$$P\{\text{false acquisition}\} = P\{M \text{ or greater transitions out of } N \mid \text{incorrect phase}\}$$

$$= \sum_{K=M}^{N} \binom{N}{K} (.5)^N$$

(163)
where $P_e$ is the error probability for the one half bit detections made for the "M-out-of-N" detector. For any value of $P_e$, a value of $M$ and $N$ can be calculated to insure a high probability of acquisition and a low probability of false acquisition. Note that these results assume that bit errors occur independently.

We first simulated the performance of the digital receiver with split-phase data using a "11-out-of-14" detector and compared the results to the performance of the digital receiver with NRZ data (Figure 77 and Figure 78). For $P_e = 0.1$, this threshold setting resulted in $P\{\text{acquisition}\} = 0.9559$ and $P\{\text{false acquisition}\} = 0.0288$. The standard deviation of bit jitter was almost identical except for low signal-to-noise ratios where the bit jitter is lower for split-phase data. Intuitively the bit synchronization algorithm should perform better for split-phase data because there are more transitions in the split-phase data.

The bit error rate is significantly higher at low signal-to-noise ratios for split-phase data. This effect is caused by the phase indicator slipping $180^\circ$ out of phase which in turn causes bursts of bit errors. On the average the phase indicator would slip $180^\circ$ out of phase once in 315 bits. To reduce the likelihood of phase slips it was decided to use a larger number of bits (i.e., larger $N$) in the decision algorithm. With $P_e = 0.1$, a threshold setting of "66-out-of-90" results in $P\{\text{acquisition}\} = 9.99998 \times 10^{-1}$ and $P\{\text{false acquisition}\} = 5.451 \times 10^{-6}$. On the average, the phase indicator would slip $180^\circ$ out of phase once in $9 \times 10^7$ bits. We simulated the performance of the
digital receiver with this decision criterion and compared the results to the performance of the receiver with NRZ data (Figure 79 and Figure 80). The rms bit jitter for both cases is nearly identical. The bit error rate for split-phase data is slightly lower except at 0 dB. We observed that the bit synchronizer slipped 180° out of phase causing bursts of bit errors much more often than Equations (163) and (164) predict. This apparently results from the bit errors not being independent. When the carrier or bit synchronization error is large, the digital receiver makes a sequence of bit errors. The bit errors are dependent, and thus Equations (163) and (164) do not hold. However, for signal-to-noise ratios greater than 3 dB, Equations (163) and (164) do accurately describe receiver performance. The conclusion is that, with split-phase data, large values of M must be used to get near optimum performance. The only bad effect of increasing M is that it increases acquisition time.
COSTAS LOOP BANDWIDTH = 0.0625 Hz
BIT SYNC LOOP TIME CONSTANT = 10 SEC
BIT RATE - NRZ = 10 BITS/SEC
BIT RATE - SPLIT-PHASE = 5 BITS/SEC
SAMPLE PERIOD = 0.005 SEC
SIMULATION LENGTH = 10,000 BITS
NRZ DATA
SPLIT-PHASE DATA
66-OUT-OF-99 DETECTOR FOR SPLIT-PHASE

BIT ERROR RATE - NRZ AND SPLIT-PHASE

Figure 79
Figure 80

133
Automatic Gain Control - Bit Synchronizer

In Section 2.5 we investigated several techniques for automatic gain control. We determined that the bandpass limiter was the preferred technique for use with a phase-locked loop, and in Section 3.1.6 we applied this technique to the Costas loop. In this section we develop an AGC technique for the bit synchronization loop.

The input sample to the bit synchronization loop is described by Equation (165) in terms of the quadrature components and the Costas loop phase estimate.

\[ V = x \cos \hat{\theta} + y \sin \hat{\theta} \]  

(165)

By applying the techniques developed in Section 2.5, the equation for the input signal with a bandpass limiter may be written as Equation (166).

\[ V = \frac{x}{\sqrt{x^2 + y^2}} \cos \hat{\theta} + \frac{y}{\sqrt{x^2 + y^2}} \sin \hat{\theta} \]  

(166)

Equation (166) should be evaluated using integer arithmetic to reduce computation time. However, there is no integer square root function available on the CDC 3200. The bandpass limiter, therefore, cannot be used for AGC in the bit synchronization loop.

Since we could not use a bandpass limiter, we determined that the most efficient technique for AGC would be a hard limiter at the input of the bit synchronizer. The input samples, V, are hard limited to \( \pm A \) volts. A curve of standard deviation of bit jitter for the sample storage bit synchronizer with and without AGC is given in Figure 81. Addition of AGC has significantly reduced bit jitter. A curve comparing bit error rates for the original digital receiver and the digital receiver with sample storage bit synchronizer and AGC in both the Costas loop and bit synchronization loop is given in Figure 82. Note that digital receiver performance is slightly better with the improved bit synchronization technique and AGC.
DIGITAL RECEIVER STUDY
AND IMPLEMENTATION

1 JUNE 1972

COSTA LOOP BANDWIDTH - 0.75 Hz
BIT SYNC LOOP TIME CONSTANT - 100 BIT PERIODS
BIT RATE - 10 BITS/SEC
SAMPLE PERIOD - 0.005 SEC
SIMULATION LENGTH - 10,000 BITS

STANDARD DEVIATION OF BIT JITTER - AGC AND NO AGC

\[ \frac{E}{N_0} \text{ dB} \]

BIT SYNC WITH SAMPLE STORAGE

BIT SYNC WITH SAMPLE STORAGE AND AGC

STANDARD DEVIATION OF BIT JITTER - FRACTION OF BIT PERIOD

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY • EAST
COSTAS LOOP BANDWIDTH = 0.75 Hz
BIT SYNC LOOP TIME CONSTANT = 100 BIT PERIODS
BIT RATE = 10 BITS/SEC
SAMPLE PERIOD = 0.005 SEC
SIMULATION LENGTH = 10,000 BITS
DIGITAL RECEIVER WITH SAMPLE STORAGE BIT SYNCHRONIZER AND AGC
DIGITAL RECEIVER WITHOUT SAMPLE STORAGE BIT SYNC

Figure 82

DIGITAL RECEIVER WITH SAMPLE STORAGE BIT SYNCHRONIZER AND AGC

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
3.1.8 Development of Integrated Program

In this section we describe the various computer programs developed for demodulating PSK data. We describe the philosophy used in developing the programs, and give instructions for using the programs.

NRZ Data with Hard Line Bit Synchronization

We developed an algorithm for demodulating PSK signals with an NRZ data format for use on the MDAC CDC 6600 hybrid computer facilities. This program was optimized for use with a signal that has hard line bit synchronization available. The hard line bit synchronization signal changes state from 0 to 10 volts to indicate the end of a bit period. All operations are done with floating point arithmetic. The computer program for hard line bit synchronization is shown in Figure 83. The program variables are defined as follows:

- SR = number of samples per bit
- BR = number of bits per second
- H = interval between samples
- BWN3DB = noise bandwidth of Costas loop
- BW = \(\omega_0\) of Costas loop
- AMP = A is the peak input signal amplitude without noise
- ZETA = \(\zeta\)
- AK = K from Costas loop
- BK = a from Costas loop
- TX = the correlator output
- IOUT = bit stream, 0 to 1
- X and Y are the input quadrature components
- AJ = the hard line bit sync signal 0 to 10 volts
- Y1 = the Costas loop phase output range \(\pm \pi/2\)
- ADC = output of analog to digital converter
\[ \pi = 3.14159 \]
\[ \pi_2 = 2 \cdot \pi \]
\[ SR = 20 \]
\[ BR = 10 \]
\[ H = 1/(BR \cdot SR) \]
\[ HI = 0.5 \cdot H \]
\[ BW_{3DB} = 0.5 \]
\[ BW = (4 \cdot BW_{3DB})/(2 \cdot ZETA + 1/(2 \cdot ZETA)) \]
\[ AMP = 1 \]
\[ ZETA = \sqrt{2}/(2) \]
\[ AK = 8 \cdot ZETA \cdot BW/AMP \cdot 2 \]
\[ BK = BW/(2 \cdot ZETA) \]
\[ DK = H \cdot AK \]
\[ CK = H \cdot BK \cdot AK \]
\[ TX = 0 \]
\[ X1 = 0 \]
\[ X2 = 0 \]
\[ IOUT = 0 \]

101 \[ X = ADC(01) \]
\[ Y = ADC(02) \]
\[ AJ = ADC(03) \]
\[ TEMP = (X + 2 - Y + 2) \cdot \sin(2 \cdot X1) + X \cdot \cos(2 \cdot X1) \]
\[ X1 = X1 + H \cdot X2 - TEMP \cdot DK \]
\[ X2 = X2 - TEMP \cdot CK \]
\[ RJN = X1/\pi + \text{SIGN}(0.5, X1) \]
\[ NNN = RJN \]
\[ QB = NNN \]
\[ Y1 = X1 - QB \cdot \pi \]
\[ TX = TX + HI \cdot (X \cdot \cos(X1) - Y \cdot \sin(X1)) \]
\[ IF = (AJ \text{ LT } 5) \text{ GO TO 60} \]
\[ IOUT = 0 \]
\[ IF (TX \text{ GT } 0.0) IOUT = 1 \]
\[ TX = 0 \]

60 \[ DAC(01) = Y1 \]
\[ DAC(02) = IOUT \]
\[ \text{GO TO 101} \]

COSTAS LOOP WITH
HARD LINE SYNC

138
DAC = output of digital to analog converter

The program inputs are X and Y, the quadrature components, and AJ, the hard line bit sync signal. X and Y must be normalized to an input amplitude of 1 volt in the absence of noise. Y1, the Costas loop phase, with a range of $\pm \frac{\pi}{2}$, and the detected bit IOUT, 0 or 1, are the outputs from the program.

NRZ Data with Data Derived Bit Synchronization

We developed a digital receiver optimized for use with the Goddard CDC 3200 computer. The digital receiver uses two GSFC subroutines for input and output of data on the CDC 3200 hybrid computer facilities. The subroutine for data input is INDATA (INT, NPTS). INT is an array dimensioned NPTS + 2. NPTS is the number of input samples buffered into the computer between computations. NPTS is used to establish the sampling rate. The first two elements of INT are the quadrature component samples. The subroutine for data output is DISTWO (NOUT). NOUT is a 2 dimensional array whose contents are displayed on the brush recorder. The contents of NOUT must be positive with a range of 0 - 1000. A generalized flow chart for the digital receiver is shown in Figure 84. Because integer operations on the Goddard CDC 3200 computer are executed much faster than floating point operations, the MDAC Digital PSK Receiver was implemented using integer logic and arithmetic. The receiver will accept input samples generated form the IF Sampling technique or the quadrature component sampling technique. However, the programs discussed in this section assume IF Sampling.

Because the digital receiver was implemented using integer arithmetic, it was necessary to scale the program variables and constants to keep as many significant figures as possible without danger of integer overflow. (The range
FLOW CHART OF DIGITAL PSK RECEIVER
WITH DATA DERIVED SYNC

READ SIGNAL TO NOISE VOLTAGE

CHOOSE SCALING FACTORS

READ: NUMBER OF TIME CONSTANTS
FOR BIT SYNC FILTER; NUMBER OF
SAMPLES PER BIT; NUMBER OF BITS
PER SECOND; BANDWIDTH OF COSTAS
LOOP; PEAK SIGNAL VOLTAGE

DEFINE BIT SYNC AND COSTAS
LOOP CONSTANTS

GENERATE TABLE OF COSINES

INITIALIZE COSTAS LOOP STATE VARIABLES
AND BIT SYNC LOOP VARIABLES

ENTER MAIN PROGRAM

INPUT QUADRATURE COMPONENTS
FROM A/D CONVERTER

COMPUTE COSTAS PHASE ESTIMATE

COMPUTE A INTEGRAL

COMPUTE B INTEGRAL

COMPUTE C INTEGRAL

GENERATE ERROR SIGNAL

PROCESS ERROR SIGNAL

GENERATE TIMING COMMANDS

COMPUTE CORRELATOR INTEGRAL

OUTPUT DETECTED BIT STREAM

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
of integers is from $-8,388,607$ to $+8,388,607$.) Sine and cosine terms for the Costas loop equations are computed by a table lookup. A table of cosine values is stored, and sine values are calculated from the identity $\sin \theta = -\cos (\theta + 90^\circ)$.

**Digital Receiver without Sample Storage Bit Synchronization**

A listing of the digital receiver without sample storage bit synchronization but with the DC bias remover is given in Appendix IV. This receiver is the least complex version of those developed under this contract. The receiver has no AGC in either the Costas or bit synchronization loops, but does have acquisition and tracking modes which are chosen by sense switch 3. Sense switch 3 must be turned on for tracking and turned off for acquisition.

This receiver is designed to operate with the IF sampler at 244 Hz IF and the bit rate at 12.2 bits/seconds and uses the subroutine INDATA. For the subroutine INDATA, NPTS must be set to 4. The first two samples from INDATA are the signal quadrature components as described in Section 3.1.1.

For the receiver to operate efficiently without hard limiting, an estimate the expected signal-to-noise ratio in the information bandwidth must be obtained for input scaling. This value in dB, SNR, should be a conservative estimate of the signal-to-noise ratio. Using an estimate lower than the actual signal-to-noise ratio will not significantly effect performance of the receiver. On the hand, too high an estimate may result in an integer overflow causing program interrupts. The value of SNR along with the time constant, $\tau$, of the low pass filter expressed in bit periods for the DC bias remover are input on the same card in $(F20.6, I10)$ format. $\tau$ is normally set at 200 bit periods.
The next five parameters are all read from one card in (2110, 3F20.6) format. They are: N, the time constant of the bit synchronization loop for acquisition expressed in bit periods; ISR, the number of samples per bit; BR, the number of bits per second; BWN3DB, the noise bandwidth of the Costas loop for acquisition; and SP, the peak signal voltage input to the A/D converter. N is a positive integer greater than zero. N must be less than 280 or an integer overflow will result due to scaling in the program. ISR must be an integer multiple of 4. The maximum Costas loop noise bandwidth is limited by sampling theorem requirements; that is, the sampling rate must be 2 to 10 times greater than the reciprocal of the loop noise bandwidth. Setting the sampling rate a factor of 5 times the reciprocal of the noise bandwidth will prevent Costas loop stability problems. BR is set equal to the number of bits per second. SP is used to scale the quadrature components IX and IY, and should be set to the peak-to-peak signal input voltage to the A/D converter in the absence of noise. The parameters for the tracking mode are both input from one data card in (I10, F20.6) format. NB is the time constant of the bit synchronization loop in bit periods, and BWA is the Costas loop noise bandwidth. Typical parameters for the receiver operating with the IF sampler at 244 Hz IF and with a bit rate of 12.2 bits/second are listed here.

N = 10 bit periods
ISR = 20 samples/bit
BR = 12.2 bits/second
BWN3DB = 1.0 Hz
SP = 1.0 volts
NB = 280 bit periods
BWA = 0.0625 Hz
The receiver has multiple outputs which are controlled by sense switches 1 and 2. The program outputs are: the detected bit, IOBIT; the Costas loop phase estimate, IXX; one input quadrature component, INT(1); the bit synchronization error signal, IR; and the correlator integral, ITX. IOBIT is a binary signal, 0 or 1, and is scaled to 0 or 1000 for display on the brush recorder. IXX has a range of $-\frac{\pi}{2}$ to $+\frac{\pi}{2}$ and is scaled to a range of 0 to 785 for display. For INT(1), 512 on the brush recorder corresponds to 0 volts. IR is the number of samples of bit timing correction, for which 600 on the brush recorder corresponds to 0 correction. One sample of correction is an increment of 40 on the brush recorder. For ITX, ISR*15 on the brush recorder corresponds to 0 volts. Sense switch 1 must be turned off to display IOBIT and INT(1). Sense switch 1 must be turned on, and sense switch 2 must be turned off to display IOBIT and IXX. Both sense switch 1 and 2 must be turned on to display ITX and IR.

Nine input parameters together define the scaling constants, the sampling period, and the Costas and bit synchronization loop parameters for acquisition and tracking. From these parameters all scaling and input-output parameters are computed.

A listing of the digital receiver with sample storage bit synchronization and the DC bias remover is given in Appendix V. This receiver has AGC in both the Costas and bit synchronization loops, and also has acquisition and tracking modes which are chosen by sense switch 3. Sense switch 3 must be turned on for tracking and turned off for acquisition. This listing is for the receiver set to operate with the IF sampler at 244 Hz IF and the bit rate at 12.2 bits/second. The receiver uses the subroutine INDATA with NPTS set to 4. The first two samples from INDATA are the signal quadrature components.
as described earlier.

In developing this receiver we devoted considerable efforts to the scaling problems encountered in converting the Costas loop and bit synchronization loop equations from floating point to integer arithmetic. We scaled the Costas loop equations from floating point to integer arithmetic. We scaled the Costas loop equations for operation over a wide range of loop noise bandwidths from $1.18 \times 10^{-2}$ to $1.18 \times 10^{1}$ Hz. A wide range of bandwidths is necessary for the Costas loop to acquire large offset frequencies rapidly before switching to a tracking mode. The loop bandwidth changes three orders of magnitude, but one of the bandwidth related parameters changes from $10^{-6}$ to $10^{3}$ - nine orders of magnitude. In order to keep as many significant figures as possible without danger of integer overflows, we divided this bandwidth range into nine divisions.

Then we scaled the bandwidth related parameters and the Costas loop equations to keep either two or three significant figures for each of these divisions. Three significant figures are used for SNR greater than 5 dB, and two significant figures are used for SNR less than 5 dB.

The digital receiver with sample storage bit synchronization uses the same input parameters as the digital receiver without sample storage bit synchronization. Because this version of the digital receiver has AGC, the receiver does not depend on SP, the peak signal amplitude, for correct loop again. The input peak signal voltage must only be large enough to prevent A/D converter quantization problems. The AGC removes receiver dependence on the input signal amplitude. The receiver outputs are the same as those described for the digital receiver without sample storage bit synchronization.
Split-Phase Data with Data Derived Bit Synchronization

We developed a digital receiver for use with split-phase data format. This receiver is identical to the NRZ receiver except an "N-out-of-M" detector has been added to determine phase for bit synchronization (N=66 and M=90).

A listing of the digital receiver for split-phase data format is given in Appendix VI. This receiver has the DC bias remover and AGC in both the Costas and bit synchronization loops. The receiver has acquisition and tracking modes which are controlled as before by sense switch 3. This listing is for the receiver set to operate with the IF Sampler at 244 Hz IF and the bit rate at 6.1 bits/second. The receiver uses the subroutine INDATA with NPTS set to 4. The first two samples are the signal quadrature components as described in Section 3.1.1.

The digital receiver uses the same input parameters that are used in the NRZ case. However, the time constant of the bit synchronization loop is limited to 140 bit periods, and the number of samples per bit, ISR, must be an integer of multiple of 8.

The receiver has multiple outputs which are controlled by sense switches 1 and 2. The program outputs are: the demodulated bit stream, IOUTT; the Costas loop phase, IXX; the coded split-phase signal before demodulation, IOBIT; one input quadrature component, INT(2); and the bit synchronization error signal, IR. IOUTT is a binary signal, 0 or 1, and is scaled to 0 or 1000 for display on the brush recorder. All the other outputs are scaled as described for the receiver without sample storage bit synchronization. Sense switch 1 must be turned off to display IOUTT and IXX. Sense switch 1 must be turned on, and sense switch 2 must be turned off to display IOBIT and INT(2). Both sense switch 1 and 2 must be turned on to display IOUTT and IR.
Baseband Digital Receiver with Data Derived Bit Synchronization

We developed a digital receiver for use with NRZ data that has already been converted to baseband. This receiver was developed from our digital PSK receiver. Since the data is at baseband there is no need for a carrier tracking loop in the receiver. Only a bit synchronization loop and a correlator are necessary for the baseband receiver. To develop the baseband receiver we modified our digital PSK receiver by removing the Costas loop equations. The remaining receiver functions are unchanged.

A listing of the baseband digital receiver for NRZ data is given in Appendix VII. The receiver has AGC, the DC bias removes, and acquisition and tracking modes controlled as before by sense switch 3. The listing is for the baseband receiver operated with a bit rate of 12.2 bit/seconds. The input bit stream was sampled directly by the A/D converter with a sampling frequency of 976 Hz. The baseband receiver uses every fourth sample from the A/D converter effectively reducing the sampling frequency to 244 Hz (20 samples/bit).

The expected signal-to-noise ratio, and the time constant of the low pass filter in the DC bias remover are input as described previously. The next four input parameters are all read from one card in (2110, 2F20.6) format. They are: N, the number of bit periods in the time constant of the bit synchronization loop for acquisition; ISR, the number of samples per bit; BR, the number of bits per second; and SR, the peak signal input voltage. The number of bit periods in the time constant of the bit synchronization loop for tracking, NB, is read from one card in (I10) format.
The receiver has multiple outputs which are controlled by sense switches 1 and 2. The receiver outputs are: the detected bit, IOBIT; the correlator integral, ITX; the bit synchronization loop error signal, IR; and the input baseband NRZ data, INT(2). All these outputs are scaled as described earlier in this section. Sense switch 1 must be turned off to display IOBIT and INT(2). Sense switch 1 must be turned on, and sense switch 2 must be turned off to display IOBIT and IR. Both sense switch 1 and 2 must be turned on to display IOBIT and ITX.

3.1.9 Study of Acquisition Techniques

We designed the Digital PSK Receiver with the philosophy that receiver operation be totally "hands off" once the computer program is compiled. To that end, all scaling, parameter definitions, and input-output operations are performed automatically after the user inputs nine parameters on data cards. In free running acquisition, the user must decide when the receiver has locked on to the noisy signal and then switch the receiver to the tracking mode. The user must also ascertain when the receiver loses lock while tracking and then switch the receiver to the acquisition mode. To make receiver operation fully "hands off", we designed logic to indicate when the receiver is in lock and studied two automatic acquisition techniques.

Acquisition Indicator

In practical applications of phase-locked loops, an indication of lock is obtained by beating a $90^\circ$ phase shifted version of the VCO output with the input signal, low pass filtering this result, and then thresholding the low pass filter output. Unfortunately, this simple technique is not applicable to the Costas loop because of the phase modulation of the input signal. This technique, if it could be applied to the Costas loop, would also only indicate
that the carrier tracking loop is in lock. For the digital receiver, a
technique is needed that will indicate when both the carrier synchronization
and bit synchronization loops are in lock. Intuitively, the correlator out-
put from the digital receiver is a good indicator of lock. For the correlator
output to be correct, both good carrier synchronization and good bit synchro-
nization are required. In lock with no noise, the correlator output will be
either +AT or -AT. To use the correlator output as an indication of lock,
it is necessary to remove the data dependent sign changes since the magnitude
of the correlator output is the parameter of interest.

Using this philosophy, we designed an acquisition indicator. The
correlator output is passed through an absolute value nonlinearity, low pass
filtered, and thresholded to determine acquisition status. We determined the
mean and variance of the correlator output in both the in-lock and out-of-lock
cases. For the out-of-lock case, we assume that the bit timing error (no
noise) was uniformly distributed between -0.5 T and +0.5 T, and that the
Costas loop phase error was uniformly distributed between -\pi and +\pi. The mean
correlator output due to signal alone is then 0 and the variance is 5(\Delta T)^2/12. For
the in-lock case, the mean correlator output is AT, and the variance is zero. We
computed the mean, \bar{z}, and the variance, \sigma^2_z, of the signal at the output of the
absolute value nonlinearity from Equations (167) and (168) where \mu and \sigma^2 are the
mean and variance of the correlator output.

\[
\bar{z} = \frac{-\mu}{\sqrt{2\pi}} e^{-\frac{\mu^2}{2\sigma^2}} + \mu \text{ erf} \left[ \frac{\sqrt{2\mu}}{2\sigma} \right] \quad (167)
\]

\[
\sigma^2_z = \sigma^2 + \mu^2 - \bar{z}^2 \quad (168)
\]

For the in-lock case, at 0.0 dB our analysis predicted the mean and variance
of the signal at the output of the absolute value nonlinearity to be 1.05 and
0.39 respectively (the correlator output has been normalized to 1.0). A
digital simulation at 0.0 dB showed a mean of 0.97 and a variance of 0.38.
For the out-of-lock case, at 0.0 dB we predicted a mean of 0.76 and variance
of 0.33. A digital simulation of the out-of-lock case showed a mean of 0.75
and a variance of 0.29. These results validate our analysis of the acquisition
indicator performance. To decrease the variance we passed the output of the
absolute value through a low pass filter with a time constant of ten bit
periods. Our analysis and simulation show that the threshold should be set
between .9 and .99 to insure a high probability of acquisition. Typical
performance of the acquisition indicator in the absence of noise is shown in
Figure 85 along with the bit tracking and phase tracking errors. The output
fluctuates around the mean out-of-lock value (0.5) until the bit tracking
error is less than 0.1 T and the phase tracking error is less than 0.5 radians.
The output then charges up to the mean in-lock value (1.0) as the tracking
errors go to zero. A strip chart recording made at NASA Goddard with the
Digital PSK receiver showing the acquisition indicator and the Costas loop
phase is given in Figure 86. These results indicate that this technique
generates a valid indication of lock.

Stepped Bandwidth Acquisition

One simple approach for automatic acquisition is to operate the receiver
with a wide bandwidth initially and then narrow the loop bandwidths in steps
as the acquisition indicator crosses successively higher threshold values.
When the acquisition indicator drops below preset threshold values, the
loop bandwidths are widened automatically to reacquire. This technique
requires that any initial offset frequency be within the pull-in range of
the widest loop bandwidth used.
COSTAS LOOP BANDWIDTH - 0.25 Hz
BIT SYNC LOOP TIME CONSTANT - 10 BIT PERIODS
BIT RATE - 10 BITS/SEC
SAMPLE PERIOD - 0.005 SEC

ACQUISITION INDICATOR - SIMULATION

Figure 85
ACQUISITION INDICATOR - EXPERIMENTAL

COSTAS LOOP BANDWIDTH - 2.0 Hz
We experimentally investigated acquisition with this technique using a Costas loop with three bandwidths 1.0 Hz, 0.25 Hz, and 0.0625 Hz. The initial bandwidth of 1.0 Hz is reduced by a factor of 4 when a threshold of 0.90 is crossed, and is again reduced by a factor of 4 when a threshold of 0.95 is crossed. With no noise and with offset frequencies within the pull-in range of the loop, the bandwidths are reduced as desired and the receiver acquires without difficulty. However, for any acquisition technique to be practical, it must result in receiver acquisition at low signal-to-noise ratios; i.e., 0.0 to 3.0 dB. We were unable to acquire using this technique at less than $\frac{E}{N_0} = 3$ dB. The acquisition indicator would cross one threshold for a few seconds, but would then go below threshold again. At low signal-to-noise ratios the acquisition indicator caused instabilities with this technique. We investigated receiver acquisition with other loop bandwidths and threshold levels, and had similar results. Although this technique is very simple to implement, its performance at low signal-to-noise ratios was unsatisfactory.

Swept Frequency Acquisition

A common approach for acquisition in phase-locked loops is the swept frequency method discussed in Section 2.6.1 and Section 2.8.3. This approach was applied to the digital receiver for automatic acquisition. The VCO frequency of a very narrow bandwidth Costas loop is swept until the acquisition indicator crosses threshold. The frequency of the sweep when the threshold is crossed is used as the initial condition on the VCO frequency of the Costas loop in the tracking mode. If this frequency is within the pull-in range of the Costas loop, the loop will rapidly acquire. Once an accurate phase reference is established, the bit synchronization loop will also pull-in rapidly.
The swept VCO output is \( \cos \left[ \omega_0 t + \theta(t) + \phi(t) \right] \) where \( \phi(t) = \omega_1 t + \frac{1}{2} \alpha t^2 \). The initial sweep frequency is \( \omega_1 \) rad/sec, and the sweep rate is \( \alpha \) rad/sec/sec. Because the frequency offsets encountered in practice at NASA Goddard are less than 1.0 Hz, the initial frequency is usually set at \( -2\pi \) or \( -\pi \) rad/sec. The frequency is then swept to the final frequency, \( \omega_f \), either at \( +2\pi \) or \( +\pi \) rad/sec. If the Costas loop has not acquired by this time, the sweep is initialized to \( \omega_1 \) and swept again to \( \omega_f \). When the threshold is crossed at \( t_a \), VCO frequency is computed, \( \omega_a = \omega_1 + \alpha t_a \). \( \omega_a \) is then used as an initial condition on the frequency of the Costas loop. If the acquisition indicator falls below frequency threshold at any time, the Costas loop is automatically swept again.

Because the Costas loop is operating in a non-linear mode during acquisition, it is impossible to analytically determine the sweep rate to insure a high probability of acquisition. For this reason a digital simulation must be used to determine the sweep rate. In Sections 2.6 and 2.8.3 we investigated probability of acquisition for a phase-locked loop with various sweep rates using a digital simulation and compared these results to those obtained by Frazier and Page. We then investigated probability of acquisition for the digital receiver using a digital simulation. For the noise free case we found that the Costas loop would not acquire with a sweep rate which insured a probability of acquisition of 1.0 for a phase-locked loop. Both loops were operated under the same conditions; i.e., identical loop noise bandwidths and identical sample rates, with no noise. In general we found that the Costas loop must be swept almost an order of magnitude slower than a phase-locked loop to acquire. This may be attributed to basic differences in the Costas loop and the phase-locked loop. First, the Costas loop has a smaller linear range than the phase-locked loop. This is because the error signal for the linearized
Costas loop is proportional to $\sin [2(\theta - \hat{\theta})]$ whereas the error signal for the linearized phase-locked loop is proportional to $\sin(\theta - \hat{\theta})$. Second, the phase error variance for the Costas loop (Equation 169) is greater than the phase error variance for the phase-locked loop for the same loop noise bandwidth, $B$, (Equation 170).

\[
\sigma^2 = \frac{NB}{2A^2} \left[ 1 + \frac{N_W}{2A^2} \right] 
\]

\[
\sigma^2 = \frac{NB}{A^2} 
\]

$W$ = bandwidth of low pass filters in Costas loop

\[
\frac{A^2}{N_o} = \text{signal-to-noise density ratio}
\]

The Costas loop generates cross terms between the signal and noise as well as squared terms. These factors account for the differences in acquisition for the Costas and phase-locked loops.

The swept frequency technique can be used for automatic acquisition, but the Costas loop VCO frequency must be swept slowly. However, for the small uncertainty expected in frequency, the low sweep rates are not restrictive.

**Variable Sampling Rate for Acquisition**

In the phase-locked loop study, we sampled at a rate consistent with the maximum expected offset frequency. However, during normal tracking operation, the frequency at the output of the loop phase detector is much less than the offset frequency. After acquisition has occurred, the sampling rate required for tracking (at high signal-to-noise ratios) need only be consistent with the bandwidth of the tracking loop. In order to increase the efficiency of our PSK demodulator, we investigated the use of a variable sampling rate acquisition technique.
There are several problems associated with mechanizing this approach. For high sampling rates, there is insufficient time to perform all calculations between samples. Since a good phase reference is basic for all receiver functions, the Costas loop is operated alone at a high sampling rate until the signal is acquired. Then the sampling rate is lowered so that all receiver functions can be executed.

The most difficult problem with this technique is that of controlling the sampling rate from the digital computer. With the quadrature component sampling technique, it is necessary to physically change the rate at which the A/D converter samples the quadrature components. This must be done by changing the sampling frequency of the A/D converter. Thus, operation of the digital receiver is interrupted while the sampling frequency is manually reduced. This method is not attractive since it requires a special A/D converter.

The problem of controlling the sampling rate from the digital computer is easily solved using the IF sampling technique. Section 3.1.1 discussed the IF sampling technique. We showed that the digital processor can change the effective sampling rate by discarding certain samples. For example, the first two samples from S/H 1 in Figure 52 are quadrature samples of 0.97 KHz IF. The digital processor can fix the interval between samples of the quadrature components at 1.024 msec by discarding the next two samples from S/H 1 before accepting the fifth and sixth samples as the next pair of quadrature samples. The digital receiver could operate the Costas loop alone with a 1.024 msec interval between samples of the quadrature components. After the Costas loop acquires, the interval between samples of the quadrature components is lengthened to 2.048 msec.
by discarding six samples between pairs of quadrature component samples. After reducing sampling rate, all receiver functions are executed. In order to reduce the sampling rate it is necessary to change the variable NPTS in the subroutine INDATA from 4 to 8 using a sense switch. The sampling rates 1.024 msec and 2.048 msec were only used for an example of variable sampling rates at .97 KHz IF. Following the method outlined here, the user defines the sampling rates to be compatible with the IF frequency being used.

Thus, variable sampling rate acquisition appears to be an attractive technique for acquiring over wider frequency uncertainties than would normally be possible.

3.2 Digital PSK System Performance Analysis

The Digital PSK program was developed and optimized especially for the NASA Goddard CDC 3200 computer. For instance, the program was written with all integer operations to get minimum computation time with no floating point hardware on the NASA Goddard CDC 3200. It was scaled to get the maximum possible resolution (i.e. minimum quantization) without danger of a variable overflow. The program in its Fortran version has a maximum bit rate (at four samples per bit) of approximately 80 bits/second. Rewriting this program in machine language would considerably increase the maximum bit rate.

The basic parameter in the PSK system evaluation is bit error rate, however the statistics of the phase error and bit timing error are also of interest for complete system optimization and analysis. Although the optimum possible performance of a PSK demodulator is well known and presented in Equation (171) in terms of the signal power to noise density ratio, $\frac{E}{N_0}$, this is not really sufficient for comparisons since
it ignores the effect of non-perfect synchronization.

\[ P_e = \frac{1}{2} \left( 1 - \text{erf} \left( \sqrt{\frac{E}{N_0}} \right) \right) \]  

(171)

Most systems will approach this bound as the bit sync and carrier tracking bandwidths go to zero; however no system can achieve this performance for wideband carrier phase and bit rate variations. We obtain expressions which relate phase error and bit timing statistics to our system parameters. These are then related to the bit error probability of the total system. These results can then be compared with the bit error probability achieved with perfect phase and bit timing and the error probability achieved with other non-ideal systems.

3.2.1 Sampling Losses

The first E/\(N_0\) loss in a digital system comes from the finite sampling rate. The low-pass filters (or bandpass filters for IF sampling) which precede the A/D converter must be wide enough to pass the uncertainty or variations in carrier frequency. Normally it might be supposed that the bandwidths of these filters were not very critical since if they were too wide the extra noise could be filtered out digitally within the computer. However this is not completely true for two reasons. First, the limited dynamic range of an A/D converter makes the quantization error of the digital processor dependent on dynamic range and thus bandwidth of the noisy input. Second, the finite sampling rate of the A/D converter limits the degree to which the noise can be filtered out. Aliasing or spectral fold-over causes an effective increase in the input noise density. This effect is decreased by increasing the sampling frequency to several times the low-pass filter bandwidth.
The loss in $E/N_0$ due to aliasing of the noise spectrum is shown in Figure 87 for the case of a single break RC filter. This figure assumes that the bandwidth of the signal is such that it is unaffected by the filtering before A/D conversion. Note that the losses shown in Figure 87 could be reduced by using sharper cut-off filters or completely eliminated by using a perfect integrate and dump filter.

Another consideration in choosing the low pass filter bandwidth is the degradation due to intersymbol interference. Figure 88 shows the degradation in signal-to-noise ratio from intersymbol interference as a function of bandwidth and bit error rate. The best choice of low pass filter bandwidth corresponds roughly to the value for which the loss due to intersymbol interference is equal to the sampling loss. Figure 89 shows a plot of the combined intersymbol interference and sampling loss as a function of bandwidth for a probability of error of $10^{-6}$ and 20 samples/bit. This indicates that a good choice of bandwidth (for low probabilities of error and 20 samples per bit) is three to three and one half times the bit rate.

The loss in performance due to quantization error is determined by the sample $S/N$ (i.e. the $S/N$ in the low-pass filter bandwidth). Let $2A$ be the total spread of the quantization levels, $\sigma^2$ the variance of the input noise samples, and $S$ the amplitude of the input signal.

Let $A = S + 3\sigma$  \hspace{1cm} (172)

so that each sample lies within the quantization levels $\pm A$ with high probability. The distribution of the quantization noise, $n_q$, will be approximately uniform,
Figure 87

Loss in Signal-To-Noise Ratio - dB

Sampling Frequency + Low Pass Filter Bandwidth

Loss Caused by Noise Aliasing
LOSS CAUSED BY INTERSYMBOL INTERFERENCE

Figure 88

FILTER BANDWIDTH/BIT RATE

LOSS IN SIGNAL-TO-NOISE RATIO - dB

$P_E = 10^{-6}$

$P_E = 10^{-3}$

$P_E = 10^{-1}$

$P_E = 10^{-2}$
FILTER BANDWIDTH/BIT RATE

TOTAL SAMPLING LOSS

SAMPLING FREQUENCY 20/T Hz

LOSS IN SIGNAL-TO-NOISE RATIO - dB

0 1 2 3 4 5 6

Figure 89
\[ P(n_q) = \begin{cases} 
\frac{Q}{2A} & \text{for } -A/Q \leq n_q \leq A/Q \\
0 & \text{for } |n_q| > A/Q 
\end{cases} \quad (173) \]

where \( Q \) is the number of quantization levels. Thus the variance of the quantization noise \( \sigma_{n_q}^2 \) is given by Equation (174).

\[ \sigma_{n_q}^2 = \frac{A^2}{3Q^2} = \left[ \frac{s + 3\sigma}{3Q^2} \right] \quad (174) \]

With a frequency offset, the quantization noise samples will be approximately independent, sample to sample. Thus, the bandwidth of the quantization noise will be approximately a factor of 10 greater than the signal bandwidth. This implies that effective quantization noise power (in the signal bandwidth) is actually about \( 1/5 \sigma_{n_q}^2 \). We will ignore the improvements due to filtering of quantization noise and obtain an approximate expression for the total system signal-to-noise ratio \((S/N)_t\).

\[ (S/N)_t = \frac{s^2}{\sigma^2 \left( 1 + \frac{(s/\sigma + 3)^2}{3Q^2} \right)} \quad (175) \]

For \( \frac{s^2}{\sigma^2} = 100 \) and \( Q = 32 \) levels, the increase in noise is only on the order of 10% and the effect decreases with increasing \( s/\sigma \). Thus the effect of quantization is not expected to be significant for the Goddard CDC 3200 (\( Q = 2^{12} \)), although quantization noise may be significant in a special purpose hardware implementation.
Low Resolution Sampling and Processing

We experimentally investigated the effect of reducing the number of bits/word both in the A/D converter and in the digital processor. We were particularly interested in whether reducing the number of bits from 12 (the word length of the A/D converter used with the GSFC CDC 3200) to 8 would significantly degrade performance. 8 bits is particularly convenient for a hardware implementation in that 8 bit read-only memories for the sine table look-up routine are readily available. Section 5 discusses the advantage of using an 8 bit word length. Reducing the word length from 12 bits to 8 bits caused no measurable change in the probability of bit error or the phase error variance at $E_{N_0} = 3$ db. The timing error variance increased from .293 to .269 (measured in terms of time between input samples). At $E_{N_0} = 9$ db there was still no measurable degradation in bit error probability, although the phase error variance increased from $2.0 \times 10^{-4}$ to $2.1 \times 10^{-4}$ radians and the timing error variance increased from .158 to .160 sample periods.

The A/D converter can operate with fewer bits/word than the digital processor when angle modulated signals are demodulated. Experimental results indicate that hard-limiting the PSK before sampling results in approximately 1.0 db loss in output $E/N_0$. However, since A/D converter speed is not really a problem in increasing bit rate, there does not appear to be any reason for looking at low-resolution, high speed, A/D converters.

Analytical results discussed earlier in this report indicate that decreasing the number of bits/word below 6 causes a degradation in performance especially for large values of $E/N_0$. 

163
3.2.2 Losses Due to Imperfect Phase Synchronization

The loss in $E/N_o$ due to imperfect bit and carrier synchronization cannot be easily predicted or even tightly bounded without several very restrictive assumptions. This section will present techniques for predicting losses due to imperfect synchronization and compare these results with experimentally determined data.

The first step in determining the effect of phase reference error is to define the statistics of the phase error. We will assume that the sampling rate is sufficient to make the dynamics of the digital Costas loop appear continuous. The effect of sampling is accounted for by using the sample signal-to-noise ratio as the continuous loop input signal-to-noise ratio. The phase error distribution function has been shown 19 to be approximated by the right-hand side of Equation (176). $a$ and $B$ are functions of the Costas loop parameters and the input signal-to-noise ratio, and $I_{JB}(x)$ is the modified Bessel function of imaginary order and of argument $x$. For high loop signal-to-noise ratios, Equation (176) is approximately Gaussian, whereas for low signal-to-noise ratios, it approaches a uniform distribution. The probability of bit error for imperfect phase synchronization can then be expressed as in Equation (177) for the case where the Costas loop bandwidth is much less than the bit rate frequency.

\[
P(\phi) = \frac{e^{(a\cos \phi + B\phi)}}{4\pi^2 e^{-B\pi} |I_{JB}(a)|^2} \int_{\phi}^{\phi + 2\pi} e^{-Bx - a\cos x} dx \quad (176)
\]
\[ P_e = \int_{-\infty}^{\infty} \left[ 1 - \text{erf} \left( \sqrt{\frac{E}{n_0}} \cos (\phi) \right) \right] P(\phi) \, d\phi \]  

(177)

For large loop signal-to-noise ratios the Costas loop is nearly linear; thus, the phase error variance can be approximated as shown in Equation (178).

\[ \sigma_{\phi}^2 = \frac{N_0 B}{A^2} \left[ 1 + \frac{N_0 W}{2A^2} \right] \]  

(178)

$W$ is the bandwidth of the low-pass filters $G(s)$. If $G(s)$ are integrate-and-dump filters, then $W = \frac{1}{2T}$ where $T$ is the integrate time. $B$ is the bandwidth of $F(s)$, and $\frac{A^2}{N_0}$ the signal-to-noise density ratio. Equation (178) indicates an objectionable property of the Costas loop. The Costas Loop generates cross terms between the signal and noise as well as squared noise terms. These terms cause loop performance to be directly dependent on both the bandwidth of the two low-pass filters $G(s)$ and the loop filter $F(s)$ (Figure 57). As our Digital PSK software is presently configured, the bandwidth of the two low-pass filters labeled $G(s)$ is assumed to be the same as the bandwidth of the analog low-pass filters used in the quadrature component generation. By proper manipulation this allows the digital filtering operation. This simplifies the digital Costas loop implementation, but it also makes the bandwidth of the external analog filters more critical. If the low-pass filter bandwidths are too wide, the extra noise will cause signal suppression when the outputs of the low-pass filters are multiplied together. This will increase the phase variance $\sigma_{\phi}^2$ as indicated in Equation (178). Figure 90 is a plot comparing the phase variance from the Digital PSK program and the linear theoretical value given by
\[ B_n = 1 \text{ HZ} \]
\[ h = 5 \text{ ms} \]
\[ \zeta = 0.707 \]

**COSTAS LOOP PHASE ERROR VARIANCE**

Figure 90
Equation (178) The linear theoretical values for phase variance are only accurate when the signal-to-noise ratio in the bandwidth of the G(s) filters is high. Figure 91 is a plot of probability of error as a function of signal-to-noise ratio with the phase reference supplied by a Costas loop. This figure assumes Gaussian phase error with a variance given by Equation (178). To simplify analysis of losses due to imperfect phase reference, Figure 91 shows the probability of error as a function of $\frac{E}{N_0}$ and the phase filter $\sigma^2$ (assuming Gaussian phase error). Thus to determine the loss in signal-to-noise ratio, the designer first determines the phase variance either by simulation or by Equation (178) and then uses Figure 91 to determine the loss in error rate performance.

3.2.3 Losses Due To Imperfect Bit Synchronization

The analysis of bit synchronization errors is somewhat more difficult since the bit synchronization estimate is limited to one of N discrete values (where N is the number of samples/bit). It has been shown that the mean absolute value of the bit timing error is inversely proportional to the square root of the signal-to-noise ratio (in the information bandwidth) and the square-root of the time constant of the averaging filter in bit tracking loop. Although these results were obtained for analog bit synchronizers, they also hold approximately for digitally implemented bit synchronizers.

Appendix III presents an analysis of mean square timing jitter for the bit synchronization technique used in the Digital PSK program. Figure 92 compares the theoretical timing jitter rms value with that measured in the Digital PSK program for a bit synchronization loop time constant of 10 bit periods and 80 samples/bit. Figure 93 shows:
PROBABILITY OF ERROR WITH A NOISY PHASE REFERENCE

Figure 91
STANDARD DEVIATION OF BIT JITTER VERSUS SIGNAL-TO-NOISE RATIO

Figure 92
Figure 93

BIT SYNCH ERROR VERSUS SIGNAL-TO-NOISE RATIO (THEORETICAL)

$\tau = \text{EQUIVALENT LOOP TIME CONSTANT (BIT PERIODS)}$

$\tau = 10$

$\tau = 20$

$\tau = 100$
the effect of the bit tracking loop bandwidth (or time constant) on rms timing jitter $\sigma_T$. The rms timing jitter can then be approximately related to the average loss in signal-to-noise ratio as shown in Equation (179) for the high signal-to-noise ratio case. This result assumes Gaussian distributed phase error and a bit width $T$. The effect of sampling rate on rms timing jitter is shown in Figure 94 which was determined by simulation. Thus, the increase in probability of error due to imperfect bit synchronization is determined by first calculating the rms value of the bit timing error using the loop parameters and then calculating the increase in probability of error.

### 3.2.4 Performance Evaluation

The past sections have compared theoretical predictions and experimental measurements of various subsystem output statistics such as the phase error and bit timing error variance. This section will look at the total system and will compare theoretical predictions and experimental measurements of probability of error for the entire MDAC Digital PSK receiver. Since demodulation of deep space telemetry is one of the chief applications of interest, it is especially important that the MDAC Digital Receiver operate as close to the theoretical optimum as the bit timing and frequency stability will allow. It is also important that the receiver be flexible and versatile without sacrificing error rate performance.

Figure 95 is a comparison of optimum theoretical error rate performance with experimental data taken using the NASA GSFC CDC 3200
EFFECT OF SAMPLE RATE ON BIT SYNC ERROR

Figure 94
COSTAS LOOP BANDWIDTH - 0.0625 Hz
BIT SYNC LOOP TIME CONSTANT - 280 BIT PERIODS
BIT RATE - 12.2 BITS/SEC
SAMPLE PERIOD - 1/244 SEC
EXPERIMENTAL DATA
RECEIVER WITH DC BIAS REMOVER AND SAMPLE STORAGE BIT SYNC

BIT ERROR RATE - SAMPLE STORAGE BIT SYNC

Figure 95
computer. This data was taken using the sample storage bit synchronizer and NRZ data. Figure 96 presents a similar comparison except that AGC was added for this set of measurements. Comparison of Figure 95 and Figure 96 indicates that AGC does not cause a perceptible degradation in error rate performance. Note that the MDAC Digital receiver is operating within less than 2 dB of theoretical. This data was taken without optimizing the pre-sampling IF filter bandwidth. The very low IF frequency (244 Hz) was chosen to simplify obtaining IF bandwidths on the order of 100 Hz or less. The pre-sampling bandpass filter was constructed using low-pass and high-pass filters with overlapping passbands. Figure 97 shows the transfer function of the optimized IF filter used to obtain the experimental data shown in Figure 98. Note that the experimental data taken with this IF filter is within 0.5 dB of the theoretical optimum. This is within the measurement error of the experiment since $E/N_0$ can generally not be measured more accurately than 0.5 dB with standard Gaussian noise sources.

Figure 99 shows a plot of experimental error rate performance with split phase data using a "66 out of 90" phase detector algorithm. Even though this data was taken without optimizing the pre-sampling IF filter, the experimental data is within 1.0 dB of the theoretical optimum.
COSTAS LOOP BANDWIDTH - 0.0625 Hz
BIT SYNC LOOP TIME CONSTANT - 280 BIT PERIODS
BIT RATE - 12.2 BITS/SEC
SAMPLE PERIOD - 1/244 SEC
EXPERIMENTAL DATA -
RECEIVER WITH DC BIAS REMOVER,
SAMPLE STORAGE BIT SYNC, AND AGC

BIT ERROR RATE - AGC

THEORETICAL PSK

Figure 96
BANDPASS FILTER

Figure 97
COSTAS LOOP BANDWIDTH = 0.0625 Hz
BIT SYNC LOOP TIME CONSTANT = 280 BIT PERIODS
BIT RATE = 12.2 BITS/SEC
SAMPLE PERIOD = 1/244 SEC
EXPERIMENTAL DATA

Figure 98

BIT ERROR RATE - OPTIMIZED IF FILTER
DIGITAL RECEIVER STUDY
AND IMPLEMENTATION

COSTAS LOOP BANDWIDTH - 0.0625 Hz
BIT SYNC LOOP TIME CONSTANT - 140 BIT PERIODS
BIT RATE - 6.1 BITS/SEC
SAMPLE PERIOD - 1/244 SEC
EXPERIMENTAL DATA - 
RECEIVER WITH DC BIAS REMOVER,
SAMPLE STORAGE BIT SYNC, AND AGC

THEORETICAL PSK

BIT ERROR RATE SPLIT - PHASE DATA

Figure 99

178

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
4. STUDY OF NON-ANALOG SYNTHESIS TECHNIQUES

One of the reasons for studying digital signal processing is that it provides a capability for implementing certain processing techniques which would be difficult or impossible with analog techniques. In particular, techniques which cannot be represented in terms of lumped parameter elements (for example, ideal time delays, two-dimensional processing algorithms, etc.) are difficult to implement with analog hardware. The restriction to one-dimensional processing prevents the development of optimum detection and estimation algorithms with analog hardware for many cases of practical interest. This follows by noting that, in the general case, an optimal detection or estimation algorithm generates and uses a conditional probability density of the state variable. Since this density changes with time, it is at least two-dimensional and may be considerable larger if other so-called "nuisance" state variables are involved. For sampled data inputs, the conditional probability density of the state vector is determined using Bayes law.

4.1 Bayesian Estimation

We assume that at any time $t_{k+1}$ the system equation governing the evolution of the state are of the following form.

\begin{align*}
X_{k+1} &= f(X_k, W_k) \\
Z_{k+1} &= h(X_{k+1}, V_{k+1})
\end{align*}

where $X_k$ is the state vector, $V_k$ the measurement noise, $Z_k$ the measurement, and $W_k$ the state disturbance, all at $k$. 

179
Let $\mathbf{z}_{k+1} = (z_1, \ldots, z_{k+1})$ be the set of previous observations. Then by definition

$$P(X_{k+1}, Z_{k+1} \mid \mathbf{z}_k) = P(Z_{k+1} \mid \mathbf{z}_k, X_{k+1}) P(X_{k+1} \mid \mathbf{z}_k)$$

(182)

$$= \int P(Z_{k+1} \mid \mathbf{z}_k, X_{k+1}) P(X_{k+1} \mid \mathbf{z}_k, X_k) P(X_k \mid \mathbf{z}_k) dX_k$$

(183)

From Equation (180) and Equation (181), we note that

$$P(X_{k+1} \mid \mathbf{z}_k, X_k) = P(X_{k+1} \mid X_k)$$

(184)

$$P(Z_{k+1} \mid \mathbf{z}_k, X_{k+1}) = P(Z_{k+1} \mid X_{k+1})$$

(185)

Therefore

$$P(X_{k+1}, Z_{k+1} \mid \mathbf{z}_k) = \int P(Z_{k+1} \mid X_{k+1}) P(X_{k+1} \mid \mathbf{z}_k, X_k) P(X_k \mid \mathbf{z}_k) dX_k$$

(186)

Finally, we obtain the solution for $P(X_{k+1} \mid Z_{k+1})$ by noting that by definition

$$P(X_{k+1} \mid Z_{k+1}) = \frac{P(X_{k+1}, Z_{k+1} \mid \mathbf{z}_k)}{P(Z_{k+1} \mid \mathbf{z}_k)}$$

(187)

Given $P(X_{k+1} \mid Z_{k+1})$, we can compute the optimal estimate of $X_{k+1}$ using any criterion of optimality we desire. For instance, the maximum likelihood estimate would be the value of $X_{k+1}$ which maximizes $P(X_{k+1} \mid Z_{k+1})$.

To illustrate the use of this approach, we consider the problem of tracking the phase of a sine wave in noise. For this case the state vector $X_k$ contains the phase and frequency of the sine wave, that is...
\[ \mathbf{X}_{k+1} = \begin{bmatrix} a & o \\ o & b \end{bmatrix} \mathbf{X}_k + \mathbf{W}_k \]  

(188)

where

\[ \mathbf{X}_k = \begin{bmatrix} f_k \\ \theta_k \end{bmatrix} \]  

(189)

\( f_k \) is the frequency offset at time \( k \) and \( \theta_k \) is the phase offset at time \( k \).

The measurement vector is given by

\[ \mathbf{Z}_k = \begin{bmatrix} \cos(\theta_k + k\Delta T f_k) \\ -\sin(\theta_k + k\Delta T f_k) \end{bmatrix} + \mathbf{V}_k \]  

(190)

\( \Delta T \) is the time between samples and \( \mathbf{Z}_k = \begin{bmatrix} a_k \\ b_k \end{bmatrix} \).

The measurement noise and the state disturbances are modeled as bivariate gaussian random variables with zero mean and covariance matrices given by \( \text{cov}(\mathbf{W}_k) = \mathbf{Q} \) and \( \text{cov}(\mathbf{V}_k) = \mathbf{R} \).

\[
\mathbf{P}(\mathbf{X}_{k+1} | \mathbf{X}_k) = \frac{1}{(2\pi |\mathbf{Q}|)^{1/2}} \text{e}^{-\frac{1}{2} \left( \begin{array}{c} f_{k+1} - af_k \\ \theta_{k+1} - b\theta_k \end{array} \right)^T \mathbf{Q}^{-1} \left( \begin{array}{c} f_{k+1} - af_k \\ \theta_{k+1} - b\theta_k \end{array} \right)}
\]  

(191)

\[
\mathbf{P}(\mathbf{Z}_{k+1} | \mathbf{X}_{k+1}) = \frac{1}{(2\pi |\mathbf{R}|)^{1/2}} \text{e}^{-\frac{1}{2} \left( \begin{array}{c} a_{k+1} - \cos(\theta_{k+1} + k\Delta T f_{k+1}) \\ b_{k+1} + \sin(\theta_{k+1} + k\Delta T f_{k+1}) \end{array} \right)^T \mathbf{R}^{-1} \left( \begin{array}{c} a_{k+1} - \cos(\theta_{k+1} + k\Delta T f_{k+1}) \\ b_{k+1} + \sin(\theta_{k+1} + k\Delta T f_{k+1}) \end{array} \right)}
\]  

(192)
$P(X_{k+1} | Z_k)$ is available from the previous step. $P(Z_{k+1} | Z_k)$ is independent of $X_k$ and thus can be easily determined by noting that

$$P(X_{k+1} | Z_{k+1}) P(Z_{k+1} | Z_k) = P(X_{k+1}, Z_{k+1} | Z_k)$$

and integrating both sides with respect to $X_k$

$$P(Z_{k+1} | Z_k) = \int P(X_{k+1}, Z_{k+1} | Z_k) dX_{k+1}$$

Thus $P(Z_{k+1} | Z_k)$ is a constant (i.e. independent of $X_k$) picked to normalize $P(X_{k+1}, Z_k | Z_k)$. Note that for computer implementation $P(X_{k+1} | Z_k)$ is only evaluated at a finite set of values of $X_{k+1}$. Thus the integration shown in Equations (186) and (194) reduces to a double summation over the possible values of $\theta_k$ and $f_k$. Thus combining the results of Equations (186), (191), (192), and (194), we note that $P(X_k | Z_k)$ is completely specified.

To obtain the optimum Bayesian estimator for a PSK system, the state vector $X_k$ should contain the phase offset and frequency offset, the bit timing and the data bits. This would require a four dimensional state vector. In order to reduce the dimensionality of the Bayesian estimator, carrier and bit synchronization are performed separately as is shown in Figure 100. This is not an optimum Bayesian estimator since the carrier synchronization is only optimum for the leading edge of the bit, $\tau$, and the value of the bit, $S$, known exactly, and the bit synchronization is only optimum for the phase offset, $\theta$, and the frequency offset, $f$, known exactly. But, it approaches the optimum as better estimates of $\hat{\tau}$, $\hat{\theta}$, $\hat{f}$, and $\hat{S}$ are obtained.
The Bayesian carrier estimator, assuming perfect bit synchronization, is given by Equations (186) and (187) with the state vector of Equation (188) the measurement vector of Equation (190), and the conditional densities of Equations (191) and (192).

Now, the Bayesian bit estimator, assuming perfect carrier synchronization, is given by

\[
P(\tau_{k+1}, S_{k+1} | Z_{k+1}) = \frac{P(\tau_{k+1}, S_{k+1}, Z_{k+1} | Z_k)}{P(Z_{k+1} | Z_k)}
\]

(195)

and

\[
P(\tau_{k+1}, S_{k+1}, Z_{k+1} | Z_k) = P(Z_{k+1} | \tau_{k+1}, S_{k+1}) \int P(\tau_{k+1} | \tau_k) P(\tau_k, S_{k+1} | Z_k) \, d\tau_k
\]

(196)
with the state equation for the bit edge of
\[ \tau_{k+1} = c \tau_k + u_k \] (197)
where again \( \tau_k \) denotes the time and \( S_k \) is the value of the bit at time \( t_k \).

The state disturbance is modeled as a Gaussian random variable with zero mean and variance \( \sigma^2 \) which yields
\[
P(\tau_{k+1} \mid \tau_k) = \left[ 2\pi \sigma^2 \right]^{-1/2} e^{-\frac{1}{2\sigma^2}(\tau_{k+1} - c\tau_k)^2} \] (198)

For bit synchronization the measurement vector is given by
\[
Z_k = \begin{bmatrix} \cos (S_k(t, \tau_k)) \\ -\sin (S_k(t, \tau_k)) \end{bmatrix} + V_k \] (199)
and the measurement noise takes the form
\[
P(Z_{k+1} \mid \tau_{k+1}, S_{k+1}) = \left[ 2\pi \mid R \mid 1/2 \right]^{-1/2} e^{-\frac{1}{2} \left[ \begin{array}{c} a_{k+1} - \cos(S_{k+1}(t, \tau_{k+1})) \\ b_{k+1} + \sin(S_{k+1}(t, \tau_{k+1})) \end{array} \right]^T \begin{bmatrix} R & 0 \\ 0 & 1 \end{bmatrix}^{-1} \begin{array}{c} a_{k+1} - \cos(S_{k+1}(t, \tau_{k+1})) \\ b_{k+1} + \sin(S_{k+1}(t, \tau_{k+1})) \end{array}} \] (200)

The desired output, the estimate of \( S_{k+1} \), is obtained from Equation (195) by choosing the maximum of
\[
P(S_{k+1} \mid Z_{k+1}) = \int P(\tau_{k+1}, S_{k+1} \mid Z_{k+1}) \, d\tau_{k+1} \] (201)
The above formulation will yield near optimum Bayesian carrier and bit synchronization. The only thing left is to determine the parameters of the state equations, $a$, $b$, and $c$. These are obtained by specification of the bandwidths of the processes described by the state equations. Given the following form of the state equations

$$Y_{k+1} = a Y_k + W_k$$  \hspace{1cm} (202)

and taking the Z transform we obtain

$$z Y(z) = a Y(z) + W(z)$$  \hspace{1cm} (203)

or

$$Y(z) = \frac{1}{(z - a)} W(z)$$  \hspace{1cm} (204)

Then the power spectral density is given by

$$S_Y(z) = \frac{1}{(z - a)} \frac{1}{(z-1 - a)} S_W(z) = \frac{1}{(z - a)} \frac{1}{(z-1 - a)} c_W^2$$  \hspace{1cm} (205)

where $c_W^2$ is the sampled power spectral density. The mean square value of $Y$ is given as

$$\overline{Y^2(kT)} = \frac{1}{2\pi j} \int S_Y(z) \frac{dz}{z}$$  \hspace{1cm} (206)

Making the change of variables $z = \frac{1 + W}{1 - W}$ we obtain

$$\overline{Y^2(kT)} = \frac{1}{2\pi j} \int S_Y(W) \frac{2\, dW}{(1 - W^2)} = \frac{1}{2\pi j} \int \left[ \frac{2\sigma_n^2}{[1 - \alpha + (1 + \alpha) W]} \frac{dW}{[1 - \alpha - (1 + \alpha) W]} \right]$$  \hspace{1cm} (207)
This is evaluated as
\[
Y^2(kT) = \frac{\sigma_n^2}{(1 - \alpha)(1 + \alpha)}
\]  
(208)

The maximum of \(S_Y(z)\) is evaluated at \(z = 1\) to yield
\[
S_Y(z) \bigg|_{\text{max}} = S_Y(z = 1) = \frac{\sigma_n^2}{(1 - \alpha)^2}
\]  
(209)

and the bandwidth is obtained from
\[
(BW) (S_Y(z = 1)) = Y^2(kT)
\]

\[
BW = \frac{1 - \alpha}{1 + \alpha}
\]  
(210)

\[
\alpha = \frac{1 - BW}{1 + BW}
\]  
(211)

4.2 Performance Evaluation

A computer simulation of the previously derived PSK synchronization algorithms utilizing separate carrier and bit synchronization was performed to determine probability of bit error. Since the conditional densities of the phase and frequency offset in Equation (187) and of the bit edge in Equation (195) are not calculatable in closed form, we compute the conditional probabilities only for a finite set of phase, frequency offset, and the bit edge values which in turn reduces the conditional densities to discrete densities. We let the phase offset, frequency offset, and bit edge each have 10 possible values. For these divisions of the state parameters the carrier tracking conditional density of Equation (187) reduces to
where $P(Z_{k+1} \mid Z_k)$ is a normalizing constant given by the double summation over $f_{k+1}$ and $\theta_{k+1}$ of the numerator. The summations over $f_k$, $\theta_k$, $f_{k+1}$, and $\theta_{k+1}$ consist of the 10 possible values of each of these parameters.

Also, for these values of the state parameters the bit tracking conditional density of Equation (195) is

$$P(\tau_{k+1}, S_{k+1} \mid Z_{k+1}) = \frac{P(Z_{k+1} \mid \tau_{k+1}, S_{k+1}) \sum_{\tau_k} P(\tau_k, S_{k+1} \mid Z_{k})}{P(Z_{k+1} \mid Z_k)}$$

and $P(Z_{k+1} \mid Z_k)$ is the summation over $\tau_{k+1}$ of the numerator. Using Equation (212) involves knowledge of $\tau_{k+1}$ and $S_{k+1}$ and using Equation (213) involves knowledge of $f_{k+1}$ and $\theta_{k+1}$ as is shown in Figure 100. In the simulation bit synchronization is physically done first, then the estimates $\hat{\tau}$ and $\hat{S}$ calculated and finally $\hat{\tau}$ and $\hat{S}$ are fed back for carrier synchronization. The estimates $\hat{\theta}$ and $\hat{f}$ are then calculated and fed back for bit synchronization. Since these estimates are updated with every sample, there is a one sample delay of the estimates $\hat{\theta}$ and $\hat{f}$. This introduces very little error because the phase and frequency offset change slowly with respect to the sampling time.

Since it is not known beforehand where the bit edge is, observation intervals, each the length of a bit period, are formed. There are parts of two bits in each observation interval and four possible combinations, $B(t, \tau)$, of these two bits $(0,0)$, $(0, \pi)$, $(\pi,0)$, or $(\pi, \pi)$. Let
\[ p_{i,j}^{n+1}(t,\tau) = \text{Prob} \left\{ B(t,\tau) = i\pi \text{ for } nT \leq t \leq nT + \tau; B(t,\tau) = j\pi \text{ for } nT + \tau < t \leq (n+1)T \right\} \quad i,j = 0,1 \]

be the joint probabilities of \( \tau \) and \( B \) in the \((n+1)^{st}\) observation interval, then

\[ p_{i,j}^{n+1}(t) = \sum_{\tau_k} p_{i,j}^{n+1}(t,\tau_k) \]

is the probability of the bit sequence \((\pi i, \pi j)\) in this interval. The estimate of the data bit at time \( t_{k+1} \), \( \hat{S}_{k+1} \), (for \( nT \leq t_{k+1} \leq (n+1)T \)) that is used for carrier synchronization is given by

\[ \hat{S}_{k+1} = \begin{cases} 0 & \text{if } p_{0,0}^{n+1}(t_{k+1}) + p_{0,1}^{n+1}(t_{k+1}) \geq \frac{1}{2} \\ \pi & \text{otherwise} \end{cases} \]

if \( t_{k+1} \leq nT + \tau \) and

\[ \hat{S}_{k+1} = \begin{cases} 0 & \text{if } p_{0,0}^{n+1}(t_{k+1}) + p_{1,0}^{n+1}(t_{k+1}) \geq \frac{1}{2} \\ \pi & \text{otherwise} \end{cases} \]

if \( t_{k+1} > nT + \tau \). At the end of each observation interval an estimate of the bit is made. This estimate at the end of the \((n+1)^{st}\) interval is given by

\[ \hat{S}_{(n+1)T} = \begin{cases} 0 & \text{if } p_{0,0}^{n+1}((n+1)T) + p_{0,1}^{n+1}((n+1)T) \geq \frac{1}{2} \\ \pi & \text{otherwise} \end{cases} \]

since this expression contains all the observations of the \((n+1)^{st}\) bit.
At the end of each observation interval the probabilities are reinitialized as

\[ P_{0,0}^{n+1}(nT,\tau) = \frac{1}{2} \left[ P_{0,0}^n(nT,\tau) + P_{0,1}^n(nT,\tau) \right] \]

\[ P_{1,0}^{n+1}(nT,\tau) = \frac{1}{2} \left[ P_{0,1}^n(nT,\tau) + P_{1,1}^n(nT,\tau) \right] \] (219)

The estimate for \( \tau_{k+1} \) that is fed back for carrier synchronization is, from Equation (213),

\[ \hat{\tau}_{k+1} = \max \left\{ \frac{P(\tau_{k+1}, x_{k+1})}{P(\tau_{k+1}, x_{k+1})} \right\} = \max \left\{ \sum_{s_{k+1}} P(\tau_{k+1}, s_{k+1} | x_{k+1}) \right\} \] (220)

The estimates of \( f_{k+1} \) and \( \theta_{k+1} \) used for bit synchronization are obtained jointly as, from Equation (212),

\[ \left\{ \hat{f}_{k+1}, \hat{\theta}_{k+1} \right\} = \max \left\{ \frac{P(f_{k+1}, \theta_{k+1}, x_{k+1})}{P(f_{k+1}, \theta_{k+1}, x_{k+1})} \right\} \] (221)

In the simulation we let the phase offset vary from 0 to 2\( \pi \), the frequency offset from -1 to +1 Hz, and the bit timing from 0 to T, the bit period. The variances and the bandwidths of the frequency and phase offsets were set to obtain a fairly rapid acquisition time. For the frequency offset and phase offset the variances were picked to be 0.04 Hz\(^2\) and (0.02)x(2\( \pi \))\(^2\) radians\(^2\) respectively and the bandwidths 0.001 Hz and (0.0025)x(2\( \pi \)) radians.
respectively. The variance and bandwidth on the bit timing were set at essentially zero. The probability of bit error for these parameters is given in Figure 101. A typical learning curve for the bit synchronization is shown in Figure 102.

Although we have shown that this approach can be implemented at very low bit rates, the small performance gains which were obtained do not justify the difficulty involved in the implementation. Note that the regular MDAC Digital PSK receiver operates within 0.5 dB of theoretical for practical tracking bandwidths and thus the maximum possible performance gain for the Bayesian approach is less than 0.5 dB.
Figure 101

PROBABILITY OF BIT ERROR WITH BAYESIAN ESTIMATION

THEORETICAL PSK

- EXPERIMENTAL

\[ \frac{E}{N_0} \ - \ dB \]

BIT ERROR RATE

10^{-3}
10^{-2}
10^{-1}
10^{0}

0
2
4
6
8
10
TYPICAL LEARNING CURVE FOR BIT SYNCHRONIZATION

Figure 102
5. HARDWARE IMPLEMENTATION

One of the primary reasons for this study of digital signal processing was to develop computer software that would enable any NASA ground station with computer facilities to demodulate low data rate telemetry without the purchase of any additional hardware. The use of digital signal processing for very low data rates (i.e. less than 10 bits/second) was also of special interest due to the difficulty of constructing very low bandwidth circuits with analog hardware.

Since digital signal processing also has considerable utility in higher bandwidth applications, it was decided to investigate the upper data rate capability of the MDAC digital PSK receiver. To significantly increase data rate, it is necessary to use special purpose digital logic rather than a general purpose computer.

A complete hardware design of the entire MDAC PSK receiver was beyond the scope of this study and not really necessary to determine the upper bit rate capability. Thus we decided to concentrate our design effort on the phase-tracking part of the receiver. The Costas loop, rather than the bit synchronizer, was chosen for hardware design because it appeared to be the more time consuming of the two calculations. Note that the phase and bit tracking loops are essentially separate algorithms which can be performed in parallel.

This section presents results of a study to determine the best hardware realization of the digital Costas phase-locked loop depicted in block diagram form in Figure 103.

Section 5.1 describes different machine organizations for the accomplishment of the task. Section 5.2 is a detailed description of one of these
DIGITAL COSTAS LOOP

Figure 103
methods. Section 5.3 describes additional effort required to complete trade-off studies between the various methods.

5.1 Machine Organizations

The required computations can be accomplished using a single time shared arithmetic logic unit (ALU). This unit is capable of addition and subtraction. With one ALU multiplication is performed by a process of shift and add and requires as many clocks as there are bits in the multiplier.

The computations may be performed in a shorter elapsed time if two arithmetic logic units share the computation load and even faster with three arithmetic logic units. However, no further increase in speed can be obtained by addition of a fourth ALU, because it would have to wait for results from previous ALU's.

A further increase in speed can be obtained by speeding up the multiplication process through the use of look-up table multipliers which can function in an add time. Such devices are now available.

Figure 104 lists these candidate machine organizations and their operating speeds assuming Transistor-Transistor-Logic (TTL). These speeds are the rate at which a pair of samples, \( x_n \) and \( y_n \), can be processed.

Note that with 20 samples/bit, we could go up to 100 K bits/sec using 12 bit accuracy for the calculations and look-up table multipliers. Section 3.2.1 shows that 12 bit accuracy does not cause significant performance degradation.

The following section of this document describes the logic required for the three ALU case and describes its operation.

5.2 Parallel, 3 Arithmetic Logic Unit (ALU) Design
CANDIDATE SYSTEMS VERSUS SPEED

<table>
<thead>
<tr>
<th>System Type</th>
<th>Serial (1 Arithmetic Unit)</th>
<th>Serial-Parallel (2 Arithmetic Units)</th>
<th>Parallel (3 Arithmetic Units)</th>
<th>Look-Up Table Multipliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>100 KHz</td>
<td>170 KHz</td>
<td>210 KHz</td>
<td>-</td>
</tr>
<tr>
<td>Bit</td>
<td>24 Bit</td>
<td>12 Bit</td>
<td>2000 KHz</td>
<td>12 Bit</td>
</tr>
</tbody>
</table>

TIME SLOT – OPERATIONS

<table>
<thead>
<tr>
<th>SLOT</th>
<th>ALU A</th>
<th>ALU B</th>
<th>ALU C</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>$X_n^2$</td>
<td>$X_n \cdot Y_n = M$</td>
<td>$Y_n^2$</td>
</tr>
<tr>
<td>001</td>
<td></td>
<td>$X_n^2 - Y_n^2 = N$</td>
<td></td>
</tr>
<tr>
<td>010</td>
<td>$N \sin 2\pi$</td>
<td></td>
<td>MCos 2X1</td>
</tr>
<tr>
<td>011</td>
<td></td>
<td>$N \sin 2\pi \cdot M \cos 2\pi = L$</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>LHKa</td>
<td>$H \cdot X^2$</td>
<td>LHKa</td>
</tr>
<tr>
<td>101</td>
<td></td>
<td>$H \cdot X^2 + X_1 = R$</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td></td>
<td>$R + LHKa = X_1$</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td></td>
<td>$X_2^2 + LHKa = X_2 + 1$</td>
<td></td>
</tr>
</tbody>
</table>

Figure 104

Figure 105
5.2.1 General Description

Eight (8) time slots are used to complete one iteration of computations (Reference Figure 103) using the parallel (3 ALU) processor design. Figure 105 identifies the computations performed in each time slot. Two clocks are required for an addition or subtraction operation. The number of clocks required for each multiplication is 2 times the length of the multiplier. The following lists the number of clocks required during each slot:

<table>
<thead>
<tr>
<th>SLOT</th>
<th>NUMBER OF CLOCKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>22</td>
</tr>
<tr>
<td>001</td>
<td>2</td>
</tr>
<tr>
<td>010</td>
<td>22 or 28*</td>
</tr>
<tr>
<td>011</td>
<td>2</td>
</tr>
<tr>
<td>100</td>
<td>22</td>
</tr>
<tr>
<td>101</td>
<td>2</td>
</tr>
<tr>
<td>110</td>
<td>2</td>
</tr>
<tr>
<td>111</td>
<td>2</td>
</tr>
<tr>
<td>Total</td>
<td>76 or 82</td>
</tr>
</tbody>
</table>

* 22 clocks using ROM Method A and 28 clocks using ROM Method B.

The maximum clock rate is limited by the time required to perform an addition. Addition time varies with word length and adder configuration. For the configuration used in this design; the typical addition time is 36 ns. with maximum value of 48 ns. If the word lengths to be added were increased to 24 bits; this addition time increases to typically 60 ns. If the clock period is chosen as 36 ns; then the time required to process one pair of samples, \( x_n \) and \( y_n \), is \( 36 \text{ ns/clock} \times 76 \text{ clocks} = 2736 \text{ ns} \) using ROM Method A. These conditions correspond to 360 K Hz operating speed. Using ROM Method B;
a clock period of 60 ns or greater is required to allow 600 ns for accessing
the ROM during slots 111 and 000. Using ROM Method B, the time required to
process is 4920 ns.

Through this design the following word lengths were selected.

<table>
<thead>
<tr>
<th>WORD</th>
<th>LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xn</td>
<td>11 + sign</td>
</tr>
<tr>
<td>Yn</td>
<td>11 + sign</td>
</tr>
<tr>
<td>Hka</td>
<td>11</td>
</tr>
<tr>
<td>HK</td>
<td>11</td>
</tr>
<tr>
<td>H</td>
<td>5</td>
</tr>
</tbody>
</table>

All computed quantities will be rounded off to 11 bits plus sign.

5.2.2 Functional Block Diagram

Major functional entities and their interconnections are identified in
Figure 106. A brief functional description of each of these blocks follows:

Operation Sequences and Clock

The logic circuitry required to time order operations is contained
within this block. An eight (8) counter and gating to decode each state
provides the slot timing signals. The clock oscillator provides the signals
for advancing the eight (8) counter when the required feedback signal indi-
cates operations within a given state are complete. The clock output is
also used to generate the $0A$, $0B$, and $M$ signal required to sequence each
arithmetic operation.

Arithmetic Logic Units (ALU's)

The logic required to perform multiplications, additions, and subtractions
is contained within these blocks. ALU's A and C perform multiplications only
whereas ALU B must multiply, add, or accomplish two (2) types of subtraction.

Each ALU uses 3 74181, 4 bit arithmetic logic unit/function generators
connected together and with a 74182, Look Ahead Carry Generator to form a 12
bit arithmetic unit capable of performing additions or subtractions in 36 ns.

Associated with these arithmetic units are the following registers:

A Register - Stores and inputs the augend or multiplicand to the arithmetic unit. This is a 12 bit latch type register.

B Register - Stores the multiplier. This is a 12 bit parallel in parallel out shift register.

Accumulator - Stores and inputs the addend or partial products to the arithmetic units. Also used to store the arithmetic unit output. This is a 12 bit parallel in, parallel out shift register.

D Register - Stores the number of operations required during multiplication. This is a 4 bit down counter.

In addition to these registers, gating logic is required to input from multiple sources. Both 4 line to 1 line and 2 line to 1 line data selectors are used. Wherever data must be complemented prior to loading (ALU B only); 4 bit True/Complement logic elements are used. The function of all other logic elements is given by their logic symbols.

Read Only Memory (ROM)

Value of the sine and cosine of the angles stored in the X1 Register are stored in ROM's. Two methods of obtaining these values have been devised.

Method A stores 128 values of the sine and cosine in the 0 to 90° quadrant. Each value contains 8 bits plus sign. Two ROM's are used; one for storing sine values and the other containing the cosine values. Addressing is accomplished by connecting the X1 register outputs, X1_8 through X1_2, to the ROM's via 4 bit True/Complement elements. The X1_10 and X1_9 register outputs which represent the 180° and 90° bits respectively (actually the 90° and 45° bits, since the times 2 multiplication has been accomplished by
shifting each bit one place to left) are used to select whether the true or complement of the input number is used in the addressing and to select the sign bit of the output word. It should be noted that the complemented address is less than the actual angle by the value of the LSB which in this case is 0.7°.

Method B uses a single storage element (made up of 4 ROM's) and can store up to 2048 values of the sine in the first quadrant. Only 512 values are accessible due to input word limitations. The output word using this method has 12 bit accuracy \( +1 - 5/8 \) bits, however, only the 11 most significant bits are used along with a sign bit. This method breaks up the sine of the angle, \( A \) into two parts. Addressing is accomplished in the same manner as Method A. Use of 3, 4 bit address in the input eliminates the 1 bit address error when complement addresses are required. Since simultaneous addressing to obtain both the sine and cosine values is not possible; this method uses a 12 bit latch to store the sine value while the cosine value is accessed.

**Storage Registers**

The following registers are used solely for storage:

- \( X_n \) and \( Y_n \) - 12 bit latches used to store the input words. Data is strobed into this register by external control.
- \( X_1 \) and \( X_2 \) - 12 bit latches used to store the output words. Data is strobed into these registers by control signals generated within the Operation Sequences.

**Switching Bits**

A total of 27 single-pole, single-throw switches are used to set in desired values of HK, HKa, and H. One throw of each switch is connected to \(+V\) for inputting a "1" and the other to \( \text{grd.} \) for a "0".
5.2.3 Logic Diagrams

Operation Sequencer and Clock (Reference Figure 107)

Initial turn-on of power clears flip-flops 1A, 1B, and 2A. Gate 3A decodes this 000 state and outputs for use by ALU's as well as enabling the 8A gate. Internally, OR gate 6A is enabled, and it in turn enables gates 7A, 7B, and 7C. Since the initial state of flip-flop 2B is cleared (Q is high), the M output will be high until a 0B output from flip-flop 3A via gate 7A changes flip flop 2B's state. This action allows the 0B output to be gated through 7D. The 0A output is outputted at all times. These 3 outputs, 0A, 0B, and M provide the primary gating signals necessary for sequencing of operations by ALU's A, B, and C.

Following completion of operations in slot 000; the D register of all three ALU's will be decremented to the 000 state. This state is decoded and brought back to the 5C gate. The output of the 5C gate then enables the 8A gate which, via IR gates 11A and 6C, enables gate 10B. Gate 10B then passes the clock pulse to flip-flop 1A causing the 8 bit counter to change to the 001 state. This state is decoded by gate 3B which outputs the signal for use by the ALU's and enables the 8B gate. Note that the change of slot from 000 to 001 results in the A/S output going high (simultaneously the Mult. goes low). This, along with the 0B signal which is still high, connects the appropriate inputs to the arithmetic unit. The next 0B signal then strobes the adder output into the ALU B unit accumulator and enables gate 8B which enables gates 10B via gates 11A and 6C. Gate 10B passes the clock on to flip-flop 1A which changes state and the counter state becomes 010. This state is decoded by gate 3C, and a sequence identical to that described for slot 000 is underway. All other operations are identical to those described for slots 000 or 001.
Timing diagrams showing the relationship of Operation Sequences Outputs are detailed in Figure (108).

**Arithmetic Logic Units A and C (Reference Figure 109).**

ALU A and ALU C are used to multiply. During this operation, the arithmetic unit, 6A, 7A, 8A, and 9A performs addition only; therefore the control inputs to the 74181's, 6A, 7A, and 8A are hardwired to the A plus B mode. The outputs of the A Register (4A and 5A) and the Accumulator (10A and 11A) are permanently connected to the 74181 inputs. The output of the 74181 which is the sum of the inputs is strobed into the Accumulator as required by the multiply routine. The operations in the multiply routine are as follows:

- **M input** - Loads Multiplicand into A Register
  - Loads Multiplier into B Register
  - Loads Multiplier length into D Register
  - Clears the Accumulator
- **ØA Input** - If LSB of B = 1, strobes the Adder output into the Accumulator.
- **ØB Input** - Shifts the contents of Ac and B register on bit to right. Decrement D Register.

When the M input from the Operation Sequence goes high, the clocks to 5A and 4A are gated high via gate 21A. This results in the loading of the data into these latches. The data will be that stored in the Accumulator or ALU B except during time slot 000. Data Selectors, 1A, 2A, and 3A route the selected data to the latches. In addition, the M signal loads the data selected by data selectors 13A, 14A, 15A, 16A, 17A, and 18A into the B register (19A and 20A); clears the Accumulator (10A and 11A); and loads $D_0$ through $D_4$ into the D register (12A).
The 0A input produces an add st. signal out of gate 25A provided that the LSB of the multiplier stored in the B register (19A and 20A) is a logic one. This add st., via gates 21C and 22B, clocks the sum of the numbers previously stored in the A register and Accumulator (initially all zeros) into the Accumulator.

When the 0A input goes low, the 0B input goes high. This input decrements the D register and raises the \( S_0 \) input to the Accumulator (10A and 11A) and the B register (19A and 20A). With \( S_0 \) and \( S_1 \) to these shift registers at the logic one level, the clock input via gates 21B, 22B, and 22C shifts the contents one place towards QA.

The 0A and 0B operations continue until the D register is decremented to 0. This state is detected by gate 26A and the output is fed back to the Operation Sequences. In addition, the D register values of 7 or less is detected and output as Sin and Cos for use in addressing the Method B ROM. The correct sign for the product stored in the Accumulator (10A and 11A) is decoded by Ex=OR gate 24A and gate 23A during the time M input is high. This value is stored in the flip-flop 27A until the D=0 signal goes high. The D=0 signal enables gate 23A which strobes the proper sign into the Accumulator sign bit.

**Arithmetic Logic Unit B (Reference Figure 110)**

The operation of ALU B is more complex than that of ALU's A and C due to the requirement that the arithmetic unit (3A, 4A, 5A, and 6A) perform additions, subtractions, and comparisons as well as multiplication operations. The outputs of the A register (1A and 2A) and Accumulator (7A and 8A) are hardwired to the arithmetic unit inputs. The arithmetic unit outputs are connected to the Accumulator inputs via True/Complement logic (15A, 16A and 17A) and Data Selectors (13A, 14A and 18A). The Data Selectors allow the
Accumulator to be loaded from the arithmetic unit for all operations and also to be loaded from the Accumulator of ALU A for operations during slots 001, 011, and 111. Selection of the arithmetic unit outputs as inputs to the Accumulator is the function of gates 23C and 25C. Gate 23A selects the shift mode for the Accumulator. This mode is required whenever 0B and a Multi signal occur simultaneously. A clock signal is gated to the clock input of the Accumulator via gates 23B and 24B to produce the shift. Clocks are also supplied to the Accumulator via gates 26B and 24B to load the Accumulator with data from the arithmetic unit during multiply operations when the add st. is high. During other arithmetic operations, 0A loads data into the Accumulator via gates 26C and 24B. Gate 27B inhibits data loading from any source during slots 101 and 110. This is necessary since the addend is already in the Accumulator during this addition as a result of being stored there during the previous slot. During arithmetic operations other than multiply, 0B is used to load the data into the A Register and Accumulator for input to the arithmetic unit.

Control of the arithmetic unit is accomplished by the S₀, S₁, S₂, S₃, and Cₙ inputs. The following table describes this control:

<table>
<thead>
<tr>
<th>S₃</th>
<th>S₂</th>
<th>S₁</th>
<th>S₀</th>
<th>Cₙ</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>A minus B</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>A plus B</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>A minus B minus 1</td>
</tr>
</tbody>
</table>

During the multiplication operation, the A plus B code must be selected. True/Complementer logic (12A) and gates 39A, 39B, 34C, and 39D provide this code. The outputs of 12A (Y₁, Y₂, Y₃, and Y₄) are all ones when there is a logic one on the B input and a logic zero on the A input. This is the case
when multiplication slots are decoded by the Operation Sequencer. These inputs to gates 39A, 39B, 39C, and 39D and the logic zero level on the A1S input provide the proper code for addition.

During other operations, the setup of these signals is controlled by the sign bits stored in the A register (2A) and the Accumulator (8A). When signs are alike, the output of Exclusive OR gate 28B will be a logic zero and the A plus B code will be set up by 12A, gates 39A, 39B, 39C, 39D, 20B and 20A. Gates 25B, 21D, 24A, and 21B provide the proper sign bit into the Accumulator. With unlike signs, the output of 28B is a logic one. True/Complementer (12A) outputs logic zeros for $S_3$ and $S_0$ and logic ones for $S_2$ and $S_1$. Gate 20A outputs a logic one for $C_n$ if the $C_{n+4}$ output of 5A is a logic zero. The arithmetic unit $C_{n+4}$ is a logic zero when the A register input is larger than the Accumulator input. As a result the arithmetic unit performs the A minus B operation, and the True of the result is loaded into the Accumulator via 15A, 16A, and 17A. Gate 20C enables the sign of A for the sign bit of the Accumulator. If the opposite is true, the $C_n$ input will be a logic zero and the arithmetic unit performs the A minus B minus 1 operation, and the complement of the results is loaded into the Accumulator. Gate 21A via gates 24A, 21D, and 25B loads the sign of Accumulator into the Accumulator.

Exclusive OR gate 28A and gates 26A, 27C, and 25B, and flip-flop 40A provide the proper sign for the Accumulator following completion of multiplication operations.

Inputs to the A Register via data selectors, 32A, 33A, 34A, 35A, 36A, and 37A. The following code is given for the inputs:
<table>
<thead>
<tr>
<th>Slot</th>
<th>A</th>
<th>B</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>000/010</td>
<td>0</td>
<td>0</td>
<td>$X_n$</td>
</tr>
<tr>
<td>100/101/111</td>
<td>1</td>
<td>0</td>
<td>$A_c$ of ALU C</td>
</tr>
<tr>
<td>001/011/101/110</td>
<td>0</td>
<td>1</td>
<td>$X_2$</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>1</td>
<td>$X_1$</td>
</tr>
</tbody>
</table>

Gates 24C and 38A set up the required selection codes A and B.

Operation of the B register (9A and 10A) and D Register (11A) is identical to that of the register described for ALU's A and C. Data selectors 19A, 41A and 42A are required to select the proper data for loading into the B register.

Gates 22B, 22C, and 22D inhibit the $f_A$, $f_B$ and M inputs to ALU B during slot 010. This is required to inhibit operations since the ALU is not operated during this time slot.

**Sine - Cosine Read Only Memory**

The sine and cosine of two times the angle stored in the X1 output are stored in Read Only Memories (ROM's). Standard ROM's storing the sine and cosine are available. These ROM's store values for angles in the 1st quadrant (0 to $\pi/4$). Addressing the ROM is accomplished by hardwiring the $X_1$ register outputs to ROM inputs. The multiplication by two is accomplished by shifting the wiring one place to left (i.e., $X_{10}$ is wired to ROM, $\pi$ input position instead of $\pi/2$). Two methods of providing the sine and cosine of the $X_1$ angles are described in the following:

**Method A (Reference Figure 111):** Two 128 word, 8 bit ROM's (1A and cosine counterpart) are used. The $X_{10}$ and $X_0$ inputs, representing the $\pi$ and $\pi/2$ bits, are connected to gates 5A, 5B, 5C, and 5D. These gates along with OR gates 4A, 4B, 4C, and 4D and True/Complementers 2A and 3A select the True/
Complement of the input angle and decode the output sign bit according to the following:

<table>
<thead>
<tr>
<th>X₁₀</th>
<th>X₁₀</th>
<th>Quadrant</th>
<th>Sine</th>
<th>Complement</th>
<th>Output Sine</th>
<th>Output Cosine</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1st</td>
<td>True</td>
<td>Complement</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2nd</td>
<td>Complement True</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3rd</td>
<td>True</td>
<td>Complement</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>4th</td>
<td>Complement True</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Method B (Reference Figure 112) - This method uses 4 ROM's to store 2048 values of the sine in the 1st quadrant. Each value is output as a 12 bit word. The \( X_1 \) register which stores the values of the angles has only 12 bits. One bit is lost in the multiplication by two and two bits are required to identify the quadrant. Hence, only 512 values are accessible. Also, the output word resolution is reduced to 10 bits due to \( B \) register length limitations. Selecting the correct address for each quadrant for both the sine and cosine is accomplished by gates 18A, 18B, 18C, 18D, 19A, 19B, 19C, 19D, and 20A. When this selection produces an input angle that required complementing (such as sine of 120°) to obtain the proper address; exclusive OR gates, 12A, 13B, 13C, 13D, 14A, 14B, 14C, 14D, 15A, 15B, 15C and adders 5A, 6A, and 7A output the required address. Similarly, adders 8A, 9A, and 10A sum the ROM outputs to produce the 12 bit output. Latches 11A and 12A store the sine value since simultaneous addressing of sine and cosines is not possible when only the sine of an angle is stored. Gates 2A, 12B, and 20B output the correct sign bit. Level converters 16A, 16B, 16C, 16D, 16E, 16F, 17A, 17B, 17C, 17D and 17E are required to interface the TTL address 5A, 6A, and 7A with the MOS ROM's.

Input-Output Registers (Reference Figure 113)

Latches are used to store the \( X_n \) (1A and 2A) inputs and the \( Y_n \) (3A and 4A)
inputs. Data is clocked into these latches whenever the respective external clock goes high.

The output registers also use latches for storage. These registers both receive inputs from the Accumulator of ALU B. Gates 9A and 9B are used to enable input clocks during the appropriate time slot.

Parts List

Total Parts Count (exclusive of ROM's)

9 74181
3 74182
6 74198
6 74194
7 74100
7 7477
12 74153
3 74193
6 7473
4 74HB7
12 9322
15 7408
7 7410
3 7420
5 7432
2 7427
1 7425
3 7486
8 7404
27 SPST Switches

Total Parts (Method A ROM)  Total Parts (Method B ROM)

1 MM522BM  1 SK003
1 MM522BY  6 7483
4 74HB7  1 74100
1 7408  1 7477
1 7432  3 7408
1 7404  1 7425
3 8812  3 7486
3 8800  1 7404
7 3K Resistors  2 8812
8 6.8K Resistors  11 3K Resistors
218  10 Diodes
  1 5.1K Resistor
5.2.4 Cost Estimate

This design uses 134 or 141 integrated circuits (depending upon the type of ROM used). Packaging and other aspects of the mechanical design of the processor such as thermal considerations have not been completed due to lack of time, however it's estimated that 3 circuit cards will be required. Based on these quantities, the following estimate of cost to produce the processor is given:

- Integrated Circuits $520
- Switches $30
- Printed Circuit Costs $390
- Connectors $15
- Labor for Assembly (180 hrs. at $22/hr) $3,960
- Labor for Test and Checkout (120 hrs. at $22/hr) $2,640
- Labor for Design Completion (40 hrs. at $22/hr) $880

Total: $8,335

5.3 Future Effort

The design described in Section 5.2 utilizes one of many methods possible. In addition, it is recommended for follow on work that the following designs be completed in the same manner to provide meaningful speed and accuracy vs. cost trade-off data:

a) Serial Processor
b) Serial-parallel processor
c) Maximum speed processor using Look-up Table Multipliers.
References


APPENDIX I - ANDERSON, BALL, VOSS METHOD

This appendix develops a set of difference equations which simulate a second order analog filter using the Anderson, Ball, Voss method. Let $F(s)$ be defined as in Equation (I 1).

$$ F(s) = \frac{\theta(s)}{e(s)} = \frac{ks + ak}{s^2} \quad (I\ 1) $$

The differential equation of the loop will have the form given in Equation (I 2).

$$ \ddot{\theta} = k\dot{e} + aKe \quad (I\ 2) $$

Following the method developed by Anderson, Ball, and Voss, we assume that the output of the filter is known up to time $t_n$, and approximate both the solution of Equation (I 2) and the input signal in the time interval from $t_n$ to $(t_n + h)$. It is then possible to solve for the coefficients of the polynomial series for $\theta(t)$ and determine the output at time $(t_n + h)$. The input signal is approximated by the polynomial in $t$ given in Equation (I 3).

$$ e = B_1 + B_2 (t-t_n) + B_3 (t-t_n)^2 \quad (I\ 3) $$

The coefficients of Equation (I 3) can then be determined as a function of the past, present and future samples of the input signal, $e$. The results are given in Equations (I 4 - I 6).

$$ B_1 = e_n \quad (I\ 4) $$

$$ B_2 = \frac{1}{2h} (e_{n+1} - e_{n-1}) \quad (I\ 5) $$

$$ B_3 = \frac{1}{2h} (e_{n+1} - 2e_n + e_{n-1}) \quad (I\ 6) $$
The solution to Equation (I 2) is approximated by its transient response plus a polynomial in \( t \). Since the transient response of Equation (I 2) is also a polynomial in \( t \), the solution will have the form given in Equation (I 7).

\[
\theta = A_1 + A_2(t-t_n) + A_3(t-t_n)^2 + A_4(t-t_n)^3 + A_5(t-t_n)^4 \tag{I 7}
\]

Equations (I 3) and (I 7) are now substituted into Equation (I 2).

\[
2A_3 + 6A_4(t-t_n) + 12A_5(t-t_n)^2 = K \left[ B_2 + 2B_3(t-t_n) \right] + aK \left[ B_1 + B_2(t-t_n) + B_3(t-t_n)^2 \right] \tag{I 8}
\]

By equating like coefficients, solutions for \( A_3 \), \( A_4 \), and \( A_5 \) can be determined.

\[
A_3 = \frac{1}{2}KB_2 + \frac{1}{2}aKB_1 \tag{I 9}
\]

\[
A_4 = \frac{1}{3}KB_3 + \frac{1}{6}aKB_2 \tag{I 10}
\]

\[
A_5 = \frac{1}{12}aKB_3 \tag{I 11}
\]

\( A_1 \) and \( A_2 \) are determined by letting \( t = t_n \) and \( t = t_{n-1} \) in Equation (I 7).

\[
A_1 = \theta_n \tag{I 12}
\]

\[
A_2 = \frac{1}{n} \left[ - \theta_{n-1} + \theta_n + A_3h^2 - A_4h^3 + A_5h^4 \right] \tag{I 13}
\]
The final difference equation can be determined by letting \( t = t_{n+1} \) and substituting the results from Equations (I 4 - I 6), (I 9 - I 11), (I 12) and (I 13) in Equation (I 7). The difference equation is given in Equation (I 14).

\[
\theta_{n+1} = 2\theta_n - \theta_{n-1} + Kh (e_{n+1} - e_{n-1})/2 + aKh^2 e_n \\
+ aKh^2 (e_{n+1} - 2e_n + e_{n-1})/12
\]  

(I 14)

Since \( \theta_{n+1} \) depends on \( e_{n+1} \), a unit delay must be added in the feedback path.

The resulting values for \( e_{n+1} \), \( e_n \), and \( e_{n-1} \) are given in Equations (I 15 - I 17).

\[
e_{n+1} = \frac{Y}{n+1} \cos \theta_n - \frac{X}{n+1} \sin \theta_n \tag{I 15}
\]

\[
e_n = \frac{Y}{n} \cos \theta_{n-1} - \frac{X}{n} \sin \theta_{n-1} \tag{I 16}
\]

\[
e_{n-1} = \frac{Y}{n-1} \cos \theta_{n-2} - \frac{X}{n-1} \sin \theta_{n-2} \tag{I 17}
\]
APPENDIX II - EQUATIONS FOR LINEAR FILTERS

This appendix develops difference equations which simulate the analog filters specified in Equations (II 1) and (II 2). Five numerical methods were examined on the basis of their expected performance and speed.

(1) Runge - Kutta Method

The transfer function of the linear filter $H_1(s)$ is given in Equation (II 1).

$$H_1(s) = \frac{\theta(s)}{E(s)} = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0s + \omega_0^2} \quad (II \ 1)$$

The differential equation can then be written from the above transfer function

$$\ddot{\theta} + 2\zeta\omega_0\dot{\theta} + \omega_0^2\theta = \omega_0^2E \quad (II \ 2)$$

Using results from Scarborough, the numerical solution is determined.

$$F(\dot{\theta}, \theta, E) = -2\zeta\omega_0\dot{\theta} + \omega_0^2E \quad (II \ 3)$$

$$A_1 = F(\dot{\theta}_{n-2}, \theta_{n-2}, E_{n-2})$$

$$A_2 = F(\dot{\theta}_{n-2} + .5A_1, \theta_{n-2} + .5\dot{\theta}_{n-2} + .125hA_1, E_{n-1})$$

$$A_3 = F(\dot{\theta}_{n-2} + .5A_2, \theta_{n-2} + .5\dot{\theta}_{n-2} + .125hA_1, E_{n-1})$$

$$A_4 = F(\dot{\theta}_{n-2} + A_3, \theta_{n-2} + h\dot{\theta}_{n-2} + .5hA_3, E_{n})$$

$$\dot{\theta}_n = \dot{\theta}_{n-2} + (A_1 + 2A_2 + 2A_3 + A_4)/6.$$  

$$\theta_n = \theta_{n-2} + h\dot{\theta}_{n-2} + h(A_1 + A_2 + A_3)/6.$$  

The transfer function for $H_2(s)$ is given in Equation (II 4).

$$H_2(s) = \frac{\theta(s)}{E(s)} = \frac{2\zeta\omega_0s + \omega_0^2}{s^2 + 2\zeta\omega_0s + \omega_0^2} \quad (II \ 4)$$
A corresponding differential Equation (II 5) can be written. To avoid taking the
derivative of the input signal, this equation is converted into two first order
Equations (II 6) and (II 7) using the dummy variable C.

\[
\ddot{\theta} + 2\zeta\omega_0 \dot{\theta} + \omega_0^2 = 2\zeta\omega E + \omega_0^2 E \tag{II 5}
\]

\[
\frac{dc}{dt} = \omega_0^2 (E-\theta) \tag{II 6}
\]

\[
\frac{d\theta}{dt} = c + 2\zeta\omega_0(E-\theta) \tag{II 7}
\]

Using results from Scarborough 1 for the solution of first order simultaneous
differential equations, the difference Equation (II 8) was formulated.

\[
F(\theta, E) = 2\zeta\omega_0 h(E-\theta) \tag{II 8}
\]

\[
G_1 = F(\theta_{n-2}, E_{n-2})
\]

\[
A_1 = hC_{n-2} + G_1
\]

\[
B_1 = (\omega_0/2\zeta)G_1
\]

\[
G_2 = F(\theta_{n-2} + A_{1/2}, E_{n-1})
\]

\[
A_2 = h(C_{n-2} + B_{1/2}) + G_2
\]

\[
B_2 = (\omega_0/2\zeta)G_2
\]

\[
G_3 = F(\theta_{n-2} + A_{2/2}, E_{n-1})
\]

\[
A_3 = h(C_{n-2} + B_{2/2}) + G_3
\]

\[
B_3 = (\omega_0/2\zeta)G_3
\]

\[
G_4 = F(\theta_{n-2} + A_3, E_n)
\]

\[
A_4 = h(\theta_{n-2} + B_3) + G_4
\]

\[
B_4 = (\omega_0/2\zeta)G_4
\]

\[
C_n = C_{n-2} + (B_1 + 2B_2 + 2B_3 + B_4) /6.
\]

\[
\theta_n = \theta_{n-2} + (A_1 + 2A_2 + 2A_3 + A_4) /6.
\]

II-2
(2) Euler's Method

Euler's method for the solution of a \( n \)th order differential equation consists of first reducing the equation to \( n \) first order simultaneous differential equations and then solving the \( n \) equations using rectangular integration.

Equation (II 2) can be converted into two first order equations by defining the state variable \( \dot{\theta} \) and \( \theta \).

\[
\begin{align*}
\dot{\theta} &= C \\
\dot{C} &= 2\zeta \omega_o C + \omega_o^2 (E - \theta)
\end{align*}
\]

The difference equations for the implementation of Euler's Method can be determined from the above equations and are given in Equations (II 11) and (II 12).

\[
\begin{align*}
\theta_n &= \theta_{n-1} + h \dot{\theta}_{n-1} \\
\dot{\theta}_n &= h \left[ -2\zeta \omega_o \dot{\theta}_{n-1} + \omega_o^2 (E_{n-1} + \theta_{n-1}) \right] + \dot{\theta}_{n-1}
\end{align*}
\]

The difference equations for filter \( H_2(s) \) can be determined from Equations (II 6) and (II 7).

\[
\begin{align*}
F &= 2\zeta \omega_o h (E_{n-1} - \theta_{n-1}) \\
\theta_n &= \theta_{n-1} + hC_{n-1} + F \\
C_n &= C_{n-1} + (\omega_o / 2\zeta) F
\end{align*}
\]
(3) Anderson, Ball, Voss Method

We will now determine the difference equation for filter $H_1(s)$, using the Anderson, Ball, Voss method. We assume that the filter output is known up to time $t_n$ and approximate both the solution to the equation and the input signal in the time interval from $t_n$ to $(t_n + h)$. As was done in Appendix I, the input signal was assumed to have the form shown in Equation (II 14).

$$E = K_1 + K_2 (t-t_n) + K_3 (t-t_n)$$  \hspace{1cm} (II 14)

The coefficients of this equation can be evaluated as a function of the past, present, and future samples of the input signal, $E$.

$$K_1 = E_n$$ \hspace{1cm} (II 15)
$$K_2 = \frac{1}{2h} (E_{n+1} - E_{n-1})$$ \hspace{1cm} (II 16)
$$K_3 = \frac{1}{2h^2} (E_{n+1} - 2E_n + E_{n-1})$$ \hspace{1cm} (II 17)

The solution to the differential equation of the filter is approximated between time $t_n$ and $t_n + h$ by the sum of its transient response and a polynomial in $t$.

$$\theta = Ae^{-\zeta \omega_0 (t-t_n)} \cos \omega_0 \sqrt{1-\zeta^2(t-t_n)}$$
$$+ Be^{-\zeta \omega_0 (t-t_n)} \sin \omega_0 \sqrt{1-\zeta^2(t-t_n)} + C_1 + C_2 (t-t_n)$$
$$+ C_3 (t-t_n)^2$$  \hspace{1cm} (II 18)

Equations (II 18) and (II 14) are now substituted into the filter equation (II 2). After equating like coefficients, the results given in Equations (II 19 - II 21) are obtained.
\[ C_3 = K_3 \]
\[ C_2 = K_2 - 4\zeta K_3/\omega \]
\[ C_1 = K_1 - 2(\zeta/\omega) K_2 + K_3 (8\zeta^2 - 2)/\omega^2 \]

The coefficients \( A \) and \( B \) are determined by letting \( t = t_n \) and \( t = t_{n-1} \) in Equation (II 18).

\[ A = \theta_n - C_1 \]
\[ B = \left[ (\theta_n - C_1) \cos \omega_0 h + C_1 e^{-\zeta \omega_0 h} - C_2 e^{-\zeta \omega_0 h} \right. \\
\left. + C_3 h^2 e^{-\zeta \omega_0 h} - \theta_{n-1} e^{-\zeta \omega_0 h} \right] / \sin \omega_0 h \]
\[ \omega = \omega_0 \sqrt{1-\zeta^2} \]

The resulting difference equation for the loop is given in Equation (II 25).

\[ \theta_{n+1} = A e^{-\zeta \omega_0 h} \cos \omega_0 \sqrt{1-\zeta^2} h \\
+ B e^{-\zeta \omega_0 h} \sin \omega_0 \sqrt{1-\zeta^2} h + C_1 + C_2 h \]
\[ + C_3 h^2 \]

The difference equation for \( H_2(s) \) can also be determined by using the general formula for a second order filter given by Anderson, Ball, and Voss. The results for \( H_2(s) \) are given in Equations (II 26 - II 32).
**Digital Receiver Study and Implementation**

\[
A = 2e^{-\zeta \omega_o h} \cos (\omega_o h \sqrt{1-\zeta^2}) \quad (\text{II 26})
\]

\[
B = -e^{-2\zeta \omega_o h} \quad (\text{II 27})
\]

\[
C = -1/\omega^2 \quad (\text{II 28})
\]

\[
D = (1-A-B) C + (1+B)/2 + (1-B)/2 
\]

\[
F = (1-A-B) (1+2C) - (1-B) 
\]

\[
G = (1-A-B) C - .5 (1+B) + .5 (1-B) 
\]

\[
\theta_n = DE_n + FE_{n-1} + GE_{n-2} + A\theta_{n-1} + B\theta_{n-2} 
\]

\[
\theta(z) = Z(1-B-DA) + C-B+DA 
\]

\[
E(z) = \frac{Z(1-B-DA) + C-B+DA}{z^2 - 2Bz + C} 
\]

\[
A = e^{-\zeta \omega_o h} \sin \omega_o h \sqrt{1-\zeta^2} \quad (\text{II 36})
\]

(4) Z-Transform

In order to determine the difference equation using the Z-transform technique, the Z-transform of the transfer function must be determined. However, in order to get meaningful results, a hold circuit must be placed before the filter. A zero order hold was used in order to simplify the implementation. The Z-transform can then be determined as shown in Equations (II 33 - II 39).

\[
\frac{\theta(z)}{E(z)} = Z \left[ \frac{1-e^{-ST}}{s} \right] \left[ \frac{\omega_o^2}{s^2 + 2\zeta \omega_o s + \omega_o^2} \right] 
\]

\[
\frac{\theta(z)}{E(z)} = (1-z^{-1}) Z \left[ \frac{\omega_o^2}{s(s^2 + 2\zeta \omega_o s + \omega_o^2)} \right] 
\]

\[
\frac{\theta(z)}{E(z)} = \frac{Z(1-B-DA) + C-B+DA}{z^2 - 2Bz + C} 
\]
The difference equation for the filter is given in Equation (II 40).

\[ \theta_n = 2B\theta_{n-1} - C\theta_{n-2} + (1-B-DA) E_{n-1} \]

\[ + (C-B+DA) E_{n-2} \]  

The difference equation using the Z - transform method for \( H_2(s) \) can be derived in a similar manner.

\[ \frac{\theta(z)}{E(z)} = (1-z^{-1}) Z \left[ \frac{2\zeta \omega_o s + \omega_o^2}{s(s^2 + 2\zeta \omega_o s + \omega_o^2)} \right] \]  

\[ A = e^{-\zeta \omega_o h} \]  

\[ B = \zeta / \sqrt{1-\zeta^2} \]  

\[ C = \cos \omega_o h \sqrt{1-\zeta^2} \]  

\[ D = \sin \omega_o h \sqrt{1-\zeta^2} \]  

\[ \theta_n = (1-AC + BAD) E_{n-1} + (A^2 - AC - BAD) E_{n-2} \]

\[ + (2AC) \theta_{n-1} - A^2 \theta_{n-2} \]
(5) **Tustin's Method**

Tustin's method requires that the filter transfer function $H_1(s)$ be written in terms of $(1/s)$ as shown in Equations (II 47) and (II 48).

$$\theta = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2}$$  \hspace{1cm} (II 47)

$$\theta(1 + \frac{2\zeta\omega_0}{s} + \frac{\omega_0^2}{s^2}) = \left(\frac{\omega_0^2}{s^2}\right) E$$  \hspace{1cm} (II 48)

The difference equation for $H_1(s)$ is determined by substituting the operator $h(1+z^{-1})2(1-z^{-1})$ for $(1/s)$.

$$\theta \left[ 1 + \frac{2\zeta\omega_0 h(1+z^{-1})}{2(1-z^{-1})} + \frac{\omega_0^2 h^2(1+z^{-1})^2}{4(1-z^{-1})^2} \right]$$  \hspace{1cm} (II 49)

$$= \left[ \frac{\omega_0^2 h^2(1+z^{-1})^2}{4(1-z^{-1})^2} \right] E$$

$$\theta \left[ (1-2z^{-1} + z^{-2}) + \zeta\omega_0 h (1-z^{-2}) + \omega_0^2 h^2 (1+2z^{-1} + z^{-2})/4 \right]$$  \hspace{1cm} (II 50)

$$= \omega_0^2 h^2 E (1+2z^{-1} + z^{-2})/4$$

This result can then be converted into the difference Equation (II 52).

$$K = 1 + \omega_0^2 h/4 + \zeta\omega_0 h$$  \hspace{1cm} (II 51)
\[ \theta_n = \frac{1}{K} \left[ (2 - \omega_o^2 h/2) \theta_{n-1} + (\zeta \omega_o h - 1 - \omega_o^2 h/4) \theta_{n-2} \right. \\
\left. + \omega_o^2 h E_n + (2E_{n-1} + E_{n-2})/4 \right] \quad (II 52) \]

Similar methods can be used to generate the difference equation for \( H_2(s) \). The result is given in Equation (II 54).

\[ K = 4 + 4\zeta \omega_o h + \omega_o^2 h^2 \quad (II 53) \]

\[ \theta_n = \frac{1}{K} \left[ (4\zeta \omega_o h + \omega_o^2 h^2) E_n + 2\omega_o^2 h^2 E_{n-1} \\
+ (\omega_o^2 h^2 - 4\zeta \omega_o h) E_{n-2} - (2\omega_o^2 h^2 - 8) \theta_{n-1} \right. \\
\left. - (4 - 4\zeta \omega_o h + \omega_o^2 h) \theta_{n-2} \right] \quad (II 54) \]
APPENDIX III - BIT SYNCHRONIZATION ERROR

This appendix presents an analysis of our digital bit synchronization technique and provides expressions for the output timing jitter as a function of signal-to-noise ratio and tracking bandwidth. The technique employed to determine bit synchronization error is similar to that used by Layland. A block diagram of the bit synchronization model is shown in Figure III-1.

BIT SYNCHRONIZATION MODEL

\[ F(z) = \frac{1}{2} (1 - D_n) \]

\[ F(z) = \frac{1}{2} (1 + D_n) \]

\[ F(z) = \frac{1}{2} (D_n - 1) \]

In the above diagram the variable \( D_n \) has the value -1 if a bit transition has occurred and has the value +1 if no transition has occurred. For this analysis the two states are assumed to be equally probable, and it is also assumed that there is no correlation between adjacent samples. The above figure also indicates that \( X_n \) is the input signal when a transition occurs, and that \( Y_n \) is the input signal when no transition occurs. In this analysis we are assuming that the bit synchronization loop is second order. The loop filter \( F(z) \) is determined as shown in Equations (III 1) and (III 2).

\[ F(z) = 2 \left[ (1 - e^{-gT}) \frac{K(s+a)}{s^3} \right] \]  

(III 1)
The difference Equation (III 3) relates $C_n$ to $E_n$.

$$C_n = 2C_{n-1} - C_{n-2} + AE_{n-1} + BE_{n-2}$$  (III 3)

Using this result, the difference equation relating the inputs and outputs of the bit synchronizer can be determined. The result is given in Equation (III 4).

$$C_n = \left[ 2 + .5A(D_{n-1} - 1) \right] C_{n-1} + \left[ .5B(D_{n-2} - 1) - 1 \right] C_{n-2} + .5A \left( 1 - D_{n-1} \right) X_{n-1} + .5A \left( 1 + D_{n-1} \right) Y_{n-1} + .5B \left( 1 - D_{n-2} \right) X_{n-2} + .5B \left( 1 + D_{n-2} \right) Y_{n-2}$$  (III 4)

We next square both sides of Equation (III 4) and take the expected value. In order to simplify the results the assumptions given in Equations (III 5 - III 7).

$$\overline{D_n^2} = 1$$  (III 5)

$$\overline{D_n} = 0$$  (III 6)

$$\overline{D_n D_{n-1}} = 0$$  (III 7)

After applying the above operations to Equation (III 4), Equation (III 8) results.

$$(-4 + 2A - .5A^2 - .5B^2 - B) C_n^2 = (A-4) \overline{Cn C_{n-1}}$$

$$+ (2B - .5AB) \left( \overline{C_n X_{n-1}} (1-D_{n-1}) + \overline{C_n Y_{n-1}} (1+D_{n-1}) \right) + (2A - B) \overline{C_n Y_n}$$

$$+ C_n \overline{C_{n-1} (D_{n-1} - 1)} + (2A - B) \overline{C_n Y_n}$$  (III 8)

$$\left( 2A - B^2 - B^2 - B \right) \overline{C_n X_n} + .5(A^2 + B^2) \left( \overline{X_n^2 + Y_n^2} \right) + .5AB \left( \overline{X_n X_{n-1}} + \overline{Y_n Y_{n-1}} \right) + AB \overline{X_n Y_{n-1}}$$
This equation can be further simplified by using Equation (III 4) to solve for the first five terms on the right side of Equation (III 5). This is accomplished by multiplying both sides of Equation (III 4) by the appropriate quantity and averaging. The results are given in Equations (III 9 - III 14).

\[
\frac{C_{n}C_{n-1}(D_{n-1})}{C_{n}C_{n-1}} = \left[ (3A-4) \frac{C_{n}^{2}}{C_{n}^{2}} - 3A \frac{C_{n}X_{n}}{C_{n}n} + A \frac{C_{n}Y_{n}}{C_{n}n} - B \frac{C_{n}X_{n-1}(1-D_{n-1})}{C_{n}n-1(1+D_{n-1})} - B \frac{C_{n}Y_{n-1}(1+D_{n-1})}{C_{n}n-1(1+D_{n-1})} \right] / (B+4)
\]

\[
\frac{C_{n}C_{n-1}}{C_{n}C_{n-1}} = \left[ (4 -A + .5 AB) \frac{C_{n}^{2}}{C_{n}^{2}} + (A - .5AB) \frac{C_{n}X_{n}}{C_{n}n} + (A + .5AB) \frac{C_{n}Y_{n}}{C_{n}n} + B \frac{C_{n}X_{n-1}(1 - D_{n-1})}{C_{n}n-1(1+D_{n-1})} + B \frac{C_{n}Y_{n-1}(1+D_{n-1})}{C_{n}n-1(1+D_{n-1})} \right] / (B+4)
\]

\[
C_{n}X_{n} = .5A (X_{n-1} + X_{n-1}n) + Y_{n-1}
\]

\[
C_{n}Y_{n} = .5A (Y_{n-1}X_{n-1} + Y_{n-1}n) + Y_{n-1}
\]

\[
C_{n}X_{n-1}(1-D_{n-1}) = (A - .5A^2 + .5B) (X_{n-1}X_{n-1} + X_{n-1}Y_{n-1}) + A X_{n}^2
\]

\[
C_{n}Y_{n-1}(1+D_{n-1}) = (A + .5B) (Y_{n-1}Y_{n-1} + X_{n-1}Y_{n-1}) + A Y_{n}^2
\]

The above equations are then substituted into Equation (III 8). The result is given in Equations (III 15 - III 20).

\[
\frac{C_{n}^2}{C_{n}} = \left[ K_{2}(X_{n}^2 + Y_{n}^2) + K_{3}Y_{n}Y_{n-1} + K_{4}X_{n}X_{n-1} + K_{5}X_{n}Y_{n-1} \right] / K_{1}
\]
\[ K_1 = -A^2 - 3B^2 - 4AB + 0.5A^2B - 0.5B^3 \]  
(III 16)

\[ K_2 = 2A^2 + 2B^2 + 4AB - 0.5A^2B + 0.5B^3 \]  
(III 17)

\[ K_3 = 2A^2 + 2B^2 + 4AB - 0.5AB^2 + 0.5A^3 \]  
(III 18)

\[ K_4 = 2A^2 + 2B^2 + 4AB - 4A^2B - 2.5AB^2 - 1.5A^3 - 0.5AB^3 + 0.5A^2B \]  
(III 19)

\[ K_5 = 4A^2 + 4B^2 + 8AB - 3AB^2 - 4A^2B - A^3 + 0.5A^3B - 0.5AB^3 \]  
(III 20)

In order to determine the bit jitter variance, the various cross products on the right side of Equation (III 15) must be evaluated. Once the cross products are evaluated Equation (III 15) provides an expression for tracking performance of the bit synchronization loop.

Figure III-2 will aid in evaluating the cross products.

**BIT SYNCHRONIZATION INTEGRALS**

The error signal is determined by integrating over the limits indicated above, and then evaluating Equation (III 21).

\[ E = (I_1 + I_2)^2 - (I_2 + I_3)^2 \]  
(III 21)

If \( B_{n-1} = -1 \), the error signal will have the form given in Equation (III 22). The random variables \( n_1 \), \( n_2 \), and \( n_3 \) are uncorrelated, have a mean value of zero, and a variance of \( \frac{1}{2} \sigma^2 \).
If both sides of Equation (III 22) are squared and averaged, Equations (III 23 - III 24) are obtained.

\[
\overline{X_n^2} = \frac{(3\sigma^2 + 3\sigma^4)}{16 V^4}
\]

\[
\overline{X_n^2} = \frac{(6S + 3)}{64S^2}
\]

\[
S = \frac{v^2}{2\sigma^2}
\]

If \(B_nB_{n-1} = 1\), the value of \(\overline{Y_n^2}\) can be derived in a similar manner.

\[
\overline{Y_n^2} = \frac{(8S + 3)}{64S^2}
\]

The cross product terms can be determined in a similar manner. If \(B_n = 1\), \(B_{n-1} = -1\), and \(B_{n-2} = 1\), the value of \(\overline{XX_{n-1}}\) can be determined as shown in Equations (III 26 - III 27).

The random variables \(n_1, n_2, n_3, n_4\), and \(n_5\) are uncorrelated, have a mean value of zero, and a variance of \(\frac{1}{2}\sigma^2\).

\[
\overline{XX_{n-1}} = \frac{[(.5V + n_1 + n_2)^2 - (.5V + n_2 + n_3)^2] - (.5V + n_3 + n_4)^2}{16V^4}
\]

\[
\overline{XX_{n-1}} = \frac{-(2S + 1)}{128S^2}
\]

The other cross product terms can be determined in a similar manner.

\[
\overline{YY_{n-1}} = \frac{-(8S + 1)}{128S^2}
\]

\[
\overline{XY_{n-1}} = \frac{-(4S + 1)}{128S^2}
\]
A graph of Equation (III 15) as a function of \( S \) is shown in Figure III-3 where it is compared to equivalent simulation results.

**STANDARD DEVIATION OF BIT JITTER VERSUS SIGNAL-TO-NOISE RATIO**

![Graph showing standard deviation of bit jitter versus signal-to-noise ratio](image)

**FIGURE III-3**
APPENDIX IV

LISTING OF DIGITAL RECEIVER WITHOUT SAMPLE STORAGE BIT SYNCHRONIZATION

IV-1
PROGRAM BIT
COMMON ICS

C MCDONNELL DOUGLAS DIGITAL PSK RECEIVER
DIMENSION ICS(650)
DIMENSION INT(100)
DIMENSION NOUT(2)
NPTS=4

C SNR IS SIGNAL TO NOISE RATIO IN DB
C NTAU IS NUMBER OF BIT PERIODS IN TIME CONSTANT OF LPF IN DC BIAS REMOVER

119 FORMAT (F20.6,110)
120 IF (SNR.LT.-10.00) 120,121
121 IEXP=1
GO TO 1c4
122 IEXP=2
GO TO 1c4
123 IEXP=3
124 IEXP=IEXP+2
IEXP=IEXP+3
I=10**IEXP
IS=10**IEXP
I6=10**IEXP
I3J=I3/I0
PI=3.14159
PI2=2*PI
IFI=3/42
IPI2=2*IF1
IPI4=IF1/2

C N IS NUMBER OF BIT PERIODS IN TIME CONSTANT OF BIT SYNC FILTER FOR ACQUISITION
C ISR IS THE NUMBER OF SAMPLES PER BIT
C BR IS THE NUMBER OF BITS PER SECOND
C BW=3DB IS NOISE BANDWIDTH IN HZ FOR COSTAS LOOP FOR ACQUISITION
C SP IS PEAK SIGNAL VOLTAGE

102 FORMAT (2110,1F20.6)
ISW=(10.*I3)/SP
LSF=100*N**2
LSF2=LSF/2
LG=4*N
KH=ISR/N+1
KI=ISR2
KJ=2*KI
KK=3*KI
IS=ISR2+1
ITP=4.*ISR
H=1./15R*BR
AMP=1.0
ZETA=50*ITP(2.)/2.0
T=1./BR

C BITW IS THE NOISE BANDWIDTH FOR THE BIT SYNCHRONIZATION LOOP

BITW=(Z.*ZETA+1./ZETA+1)/(4.*ZETA**N+1)

C BW IS OMEGA 0 IN RAD/SEC FOR COSTAS LOOP
BW=(4.*BW3DB)/(2.*ZETA+1./2.*ZETA)

C BK ANY AK ARE LOUD GAIN CONSTANS FOR THE COSTAS LOOP
B=8.*Z.TA*B*/(AMP**AMP)
```
AX=4.*BW*BW/(AMP*AMP)
DO 771 M=1,IP12
771 ICS(M)=I3*COS(IP12/IP12)
C ICS(M) IS TABLE OF COSINES
IX1=IP12
IX1=IX1+1
IX2=0
IXA=IP12
IXB=IP12
I10=IP12+1
I32=IP12/2
I3R=0
I3R=IP12
I3HT=0
I1T=0
NHTA=0
NTHA=0
ICN=0
IR=0
IA=0
ICOUNT=0
IUM=0
IIR=1
JJ=0
I1Z=0
KCOUNT=0
C CONSTANTS FOR IX1
MP=-I3/2
D=H*E
L0=0*16
MOD=ISR*BR
C CONSTANTS FOR IX2:
G=H*AK
NL=0*16
C NB IS NUMBER OF BIT PERIODS IN TIME CONSTANT OF BIT SYNC FILTER
C FOR TRACKING
C BWA IS NOISE BANDWIDTH IN HZ FOR COSTAT LOOP FOR TRACKING
READ (5,103) NB,BWA
103 FORMAT (I10,F20.6)
NSF=100*NB**2
NSF=NSF/2
NG=4*NB
BITUX=(2.*ZETA+1./(2.*ZETA))/(4.*ZETA*NB*1)
BWN=(4.*BWA)/(2.*ZETA+1./(2.*ZETA))
AN=8.*ZETA*BWN/(AMP*AMP)
DA=H*BN
ND=DA*16
GA=H*AN
NNG=GA*16
WRITE (6,500)
500 FORMAT (4X,38MICDONNELL DOUGLAS DIGITAL PSK RECEIVER)
WRITE (6,503) BR
503 FORMAT (/42X,11HB_WHT RATE IS,F10.5,2X,15HBITS PER SECOND)
WRITE (6,505) ISR
505 FORMAT (/42X,14MSAMPLE RATE IS,15,2X,15MSAMPLES PER BIT)
```
**DIGITAL RECEIVER**
**STUDY MDC E068**
**AND IMPLEMENTATION**

JUNE 1, 1972

```
WRITE (0.506)
506 FORMAT (/4X,32HACQUISITION BANDWIDTH PARAMETERS)
WRITE (0.501) BNA3DB
501 FORMAT (/3X,32HCOSTAS LOOP NOISE BANDWIDTH IS, $10.5,2X,2HZ)
WRITE (0.502) BIT3W
502 FORMAT (/3X,43HBIT SYNCHRONIZATION LOOP NOISE BANDWIDTH IS, $10.5,2X,2HZ)
WRITE (6,.507)
507 FORMAT (/4X,29HTRACKING BANDWIDTH PARAMETERS)
WRITE (6.501) BMA
WRITE (6.502) BIT3X
IAX=0
IAY=0
IAA=10000
IF (IAA.EQ.1.1) 410,411
410 IAA=10*IAA
411 IALPHA=NTAU*ISR
IFIV=IAA/ISR
C ENTRY POINT FOR X AND Y IS 101
C X AND Y ARE QUADRATURE COMPONENTS
101 CALL INDATA(INT,NPTS)
C SSMTCHF(3) ON FOR TRACKING
C SSMTCHF(3) OFF FOR ACQUISITION
GO TO (709,708) SSMTCHF(3)
706 ISF=LSF
ISF2=LSF2
IG=LG
MU=LD
MG=LLG
GO TO 790
789 ISF=NSF
ISF2=NSF2
IG=NG
MO=ND
MU=HNG
790 CONTINUE
IX=(ISQ*INT(1))/2048
IY=(ISQ*INT(2))/2048
C DC BIAS REMOVER
IAX=IAX/IALPHA+IFIV*IX/NTAU
IAY=IAY/IAY/IALPHA+IFIV*IY/NTAU
IX=IX-IAX/IAA
IY=IY-IAY/IAA
ICOUNT=ICOUNT+1
IDUM=IDUM+1
IR=0
IF (IAA.EQ.1) 609,610
609 IF (IGN.EQ.1) 78,610
610 IF (ICOUNT.GE.KH) 7,613
7 IIR=IIR-1
I=0
IA=0
ID=0
GO TO 8
78 IGN=0
```
IAX=IC
IB=0
IC=0
I=KI
IAZ=4
I=I+1
ICOUNT=KH
C COMMON TERMS FOR X1 AND X2
C FIRST TERM
S13   MA=(IX**2-IY**2)/4
      MAP=MA/13
      MB=-ICS(IXA)*ICS(IXB)
      MDP=MB/13
      MC=MAP*MDP
      MCP=MC/13
C SECOND TERM
      MF=(IX*IX)/2
      MMP=MF/13
      MN=ICS(IXA)**2
      MQ=MMP*MN
      MR=MNP*MQ
      MRP=MR/13
C COMBINE MRP AND MCP TO USE IN BOTH EQUATIONS
      MIRV=MCP+MRP
C BEGIN CALCULATION OF IX1
      ME=MU*MIRV
      MEP=ME/13
      MF=IX2*MN
      IX1=IX1+ME+MF
      L=IX1+10
      IX1=IX1-L*10
      IF (IX1.LE.0) 202,203
      202 IX1=IX1+110
      203 IF (IX1.LT.13) 204,205
      204 IXA=IPI2
          IXB=IPI1
          GO TO 236
      205 IXA=IX1/13
          IXB=IXA*IPI4
          IF (IXA.EQ.IPI2) 206,207
      206 NNN=IX1/IPI+SIGN(I32,IX1)
          NNN=NNN/13
          IXX=IX1-NN*IPI*I3
          IXA=IXX/I3
      207 IXA=IX2/13
          IXX=IX2+MF
          IXX=(IX*XCS(IXA)+IY*ICS(IXB))/13
      208 IF (ICOUNT.GE.KH) 612,100
      209 IF (I/DUH.LT.15R) 12,614
      612 IGBIT=0
IF (ITX.GT.0) 615, 616
615  I0BIT=1
616  ITX=0
I0HM=0
12  IF (I0R.GT.0) 100, 625
625  IF (ILX*+KI) 626, 627
626  I0A=IA+I0AZ
627  IF (I0R*KI,AND.ILX*+KI) 628, 629
628  I0=I0+I0AZ
629  I0C=IC+I0AZ
630  I0F=IF(I0R,KK) 632, 633
631  I0A=I0+10G
632  I0L=I0+11G
633  I0E=I0+11G
634  I0=I0+11G
635  I0=I0+11G
L1=L1/I03
L2=I0/I0G
L3=L3/I03
TER=I0R=I0E
I0H=I0*+I0G
I0H=I0F
I0R=I0A
I0=I0TH=I0TH
I0=I0TH
I0=I0TH
I0=I0TH
IF (I0B, 7000000) 701, 701
700  I0=I0TH=I0TH
701  IF (I0B, 7000000) 703, 703
703  I0C=I0C
704  GO TO 100
100  CONTINUE
C SSWTCHF(1) OFF FOR UNMODULATED BITS AND INUT SIGNAL
C SSWTCHF(2) 1 AND SSWTCHF(2) OFF FOR UNMODULATED BITS AND COSTAS LOOP PHASE
C SSWTCHF(2) 1 AND SSWTCHF(2) ON FOR CORRELATOR INTEGRAL AND BIT SYNC
C ERROR SIGNAL
GO TO (900, 901) SSWTCHF(1)
300  GO TO (300, 301) SSWTCHF(1)
301  NGT(1) = (ITX+12*ISW)/13*15
NGT(2) = 40*IR+600
GO TO 302
302  NGT(1)=I0BIT*1000
NGT(2)=I0X+1IP14/4
GO TO 302
304  NGT(1)=I0BIT*1000
NGT(2)=I0T(1)+512
302  CONTINUE
C CALL OISTWO(NOUT)
GO TO 101
END
APPENDIX V

LISTING OF DIGITAL RECEIVER WITH SAMPLE STORAGE BIT SYNCHRONIZATION
PROGRAM BIT
COMMON ICS
C MCDONNELL DOUGLAS DIGITAL PSK RECEIVER
C RECEIVER HAS DC BIAS REMOVER, AGC, SAMPLE STORAGE BIST SYNC
DIMENSION ICS(6256),LS(50),NOUT(2),INT(22)
NPS=4
DO 9 IND=1,50
9 LS(IND)=0
C SNR IS SIGNAL TO NOISE RATIO IN DB
C 'NTAU IS NUMBER OF BIT PERIODS IN TIME CONSTRAINT OF IF IN DC BIAS REMOVER
READ (5,119) SNR,NTAU
119 FORMAT (F20.6,110)
121 IF (SNR.LT.5.00) 122,123
122 IEXP=2
GO TO 124
123 IEXP=3
124 IEXP=IEXP+2
IFX=IEXP+3
I3=10**IEXP
I5=10**IEXPQ
I6=10**IEXPX
I3=13/I0
I8=133
I9=13
I10=133
I13=I3
PI=3.14159
PI2=2*PI
PI2=3.142
PI2=2*PI2
PI2=PI2
C N IS NUMBER OF BIT PERIODS IN TIME CONSTRAINT OF BIT SYNC FILTER FOR ACQUISITION
C ISR IS THE NUMBER OF SAMPLES PER BIT
C BR IS THE NUMBER OF BITS PER SECOND
C BNNUB IS NOISE BANDWIDTH IN HZ FOR COSTAS LOOP FOR ACQUISITION
C SFR IS PEAK SIGNAL VOLTAGE
READ (5,102) N,ISR,SR,BNNUB,SFR
102 FORMAT (2110,3F20.6)
ISF1=100*N**2
ISF2=ISF1/2
IG=4*N
K1=ISR/4+1
K2=ISR/2
K3=K1
K4=K1+1
K11=K1-1
K11=K1+1
K11=K1+1
I=0
IT2=ISR
H=1./ISR/BT2
AMP=1.0
ZETA=SQRT(2.)/2.0
T=1./BT2

MACDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
MDC-0648
1 JUNE 1972
DIGITAL RECEIVER STUDY
AND IMPLEMENTATION

C BITBW IS THE NOISE BANDWIDTH FOR THE BIT SYNCHRONIZATION LOOP
BITBW=(4.*ZETA+1./2.*ZETA)/4.*ZETA*NB*T)
C EW IS OMEGA IN RAD/SEC FOR COSTAS LOOP
EW=(4.*BW/300)/(2.*ZETA+1./2.*ZETA)
C BK AND AK ARE LOOP GAIN CONSTANTS FOR THE COSTAS LOOP
BK=8.*ZETA*BW/(AM+*AMP)
AK=4.*BW*JW/(AM+*AMP)
DG 771 M=14,IP12

CICS(M) IS TABLE OF COSINES

IX2=0
IXA=1IPM2
IXB=1PM4
I1U=1PM13
I32=13/2
ICR=0

C COUNT=0
I1W=0
C CONSTANT FOR IX1
MF=13/2
D=M*0K
M=K+5
C CONSTANT FOR IX2
G=K+AK
JFL=110
JFM=13
JFN=132

C BU IS NUMBER OF BIT PERIODS IN TIME CONSTANT OF BIT SYNC FILTER
C FOR TRACKING
C BN IS NOISE BANDWIDTH IN HZ FOR COSTAS LOOP FOR TRACKING
READ (5,103) NB,*BNA
103 FORMAT (11U,F20.6)
NSF=110*NB+2
NSF=NSF/2
NB=6*NB
BITUX=(4.*ZETA+1./2.*ZETA)/(4.*ZETA*NB+1.
BN=(4.*BNA)/(2.*ZETA+1./2.*ZETA)
BN=8.*ZETA*BWN/(AM+*AMP)
AN=4.*BWN/(AM+*AMP)
DA=K+B
GA=K+AN
IF (.0116.LE.BWA.AND.0,0187.GT.BWA) 740,7+1
740 NO=UA*16
DIGITAL RECEIVER STUDY AND IMPLEMENTATION

JUNE 1972

1. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

2. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

3. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

4. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

5. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

6. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

7. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

8. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

9. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

10. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

11. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

12. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

13. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

14. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

15. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

16. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

17. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

18. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

19. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

20. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

21. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

22. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

23. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

24. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

25. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

26. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

27. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

28. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

29. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

30. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

31. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

32. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

33. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

34. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

35. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

36. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

37. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

38. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

39. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

40. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

41. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

42. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

43. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

44. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

45. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

46. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

47. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

48. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

49. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

50. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

51. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

52. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

53. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

54. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

55. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

56. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

57. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

58. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

59. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

60. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

61. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

62. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

63. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

64. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

65. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

66. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

67. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

68. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

69. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

70. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

71. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

72. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

73. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

74. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

75. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

76. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

77. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

78. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

79. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

80. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

81. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

82. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

83. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

84. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

85. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

86. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

87. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

88. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

89. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

90. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

91. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

92. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

93. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

94. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

95. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

96. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

97. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

98. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

99. DIGITAL RECEIVER STUDY AND IMPLEMENTATION

100. DIGITAL RECEIVER STUDY AND IMPLEMENTATION
| 712 | MD=0*16  
|     | MG=G*I6*10  
|     | KIX=13  
|     | KIXX=10*I3  
|     | LSCAL=13  
|     | NSCA=10  
|     | IF (SNR.LT.5.00) 83,713  
|     | I36=100  
| 713 | IF (.0375*.E.BWN3DB.AND.+.118*.GT.BWN3DB) 714,715  
| 714 | MD=0*15  
|     | MG=G*I6  
|     | KIX=13  
|     | KIXX=13  
|     | LSCAL=13  
|     | NSCA=10  
|     | IF (SNR.LT.5.00) 84,715  
|     | I36=100  
| 715 | IF (.114*.E.BWN3DB.AND.+.187*.GT.BWN3DB) 716,717  
| 716 | MD=0*15  
|     | MG=G*I5  
|     | KIX=13  
|     | KIXX=13/10  
|     | LSCAL=13  
|     | NSCA=10  
|     | IF (SNR.LT.5.00) 85,717  
|     | I36=100  
| 717 | IF (.187*.E.BWN3DB.AND.+.375*.GT.BWN3DB) 718,719  
| 718 | MD=0*(15/10)  
|     | MG=G*I5  
|     | KIX=13/10  
|     | KIXX=13/10  
|     | LSCAL=13*10  
|     | NSCA=100  
|     | IF (SNR.LT.5.00) 86,719  
|     | I36=100  
| 719 | IF (.375*.E.BWN3DB.AND.+.118*.GT.BWN3DB) 720,721  
| 720 | MD=0*(15/10)  
|     | MG=G*(15/10)  
|     | KIX=13/10  
|     | KIXX=13/10  
|     | LSCAL=13*10  
|     | NSCA=100  
|     | IF (SNR.LT.5.00) 87,721  
|     | I36=100  
| 721 | IF (.118*.E.BWN3DB.AND.+.67*.GT.BWN3DB) 722,723  
| 722 | MD=0*(15/10)  
|     | MG=G*13  
|     | KIX=100  
|     | KIXX=1  
|     | LSCAL=13*13/100  
|     | NSCA=13/10  
|     | IF (SNR.LT.5.00) 730,723  
|     | I36=100  
| 730 | JFL=JFL/10  
|     | JFM=JFM/10  
|     | JFN=JFN/10  
|     | I36=100  

McDONNELL DOUGLAS ASTRONAUTICS COMPANY • EAST
DIGITAL RECEIVER STUDY AND IMPLEMENTATION

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST

10F2=ISF2
I0=1G
I34=I36
I35=I37

IF (IKW.K.EQ.KQM) 730,792
IX1=IX1/LIX
IX2=IX2/LIX
JFL=JFL/LIX
JFM=JFM/LIX
JF=JF/LIX
GO TO 790

789
MT=ND
MR=NNG
KUK=1
KIA=NIX
K1AA=N1XX
LSPAN=NLSCAL
NSCN=NN.GA

IEF=NSF2
I0=NG
I34=I3B
I35=I39

IF (IKX.EQ.KQM) 790,793
IX1=IX1*LIX
IX2=IX2/LIX
JFL=JFL/LIX
JFM=JFM/LIX
JF=JF/LIX

790
KUK=KUK
IX=(ISG*INT(111))/2448
IX=(ISG*INT(12))/2448

C DO BIAS REMVC.R

IAX=IAX-IAX/IALPHA+IFIV*IX/NTAU
IAY=IAY-IAY/IALPHA+IFIV*IY/NTAU
IX=IX-IAX/IAA
IY=IY-IAY/IAA
ICOUNT=ICOUNT+1
IUM=IUM+1

IF (IAX.EQ.0) 8,609
IF (ICOUNT.EQ.1) 78,610
610

7
IA=0
I0=0
IC=0

IF (IR.EQ.0) 8,303
IF (IR.LT.0) 303,304

303
KHJ=KJ1+IR
KHH=KHH+KII
DO 300 IJK=KHJ,KHJJ

300
IA=IA+LS(IJK)
KK=K1+IR
LS=KH
DO 301 IJK=KK2,KK
I0=1G+LS(IJK)
LS(LSS)=LS(IJK)
DIGITAL RECEIVER STUDY
AND IMPLEMENTATION

C COMMON TERMS FOR XIANO X2
C FIRST TERM
613 MAP=(IX**2-IY**2)/4/I3
       MCP=ICS(IAXA)*ICS(IX8)/I3
C SECOND TERM
       MNF=1CS(IAXA)*ICS(IX8)/I3
       MDP=MNP+MHP
       MRP=MHP+MNP/I33
C AGC FOR COSTAS LOOP
633 MA=3+IY**2)
       MAA=(IX**2+IY**2)/LSCAN
       IF (MAA.EQ.0) 169,167
C AGC FOR COSTAS LOOP
166 MAA=1
167 MEP=MT*IAXA/MM
       L=IX8/1FL
       IF (L.EQ.0) 202,203
       IX=IX8+JFL
       IF (IX1.IX1.EQ.0) 202,203
202 IX=IX8+JFL
203 IF (IX1.IX1.EQ.JFM) 204,205
204 IXA=IP12
IX=IP14
GO TO 206
205 IX=IX8+J4
       IX=IXA+IP14
       IF (IXD.EQ.IP12) 206,207
207 L=IX8/IP12
       IXD=IX8-L+IP12
286 NNN=IX1/IP14+ISUN(JFM,IX1)
       NN=NNN/JFM
       IX=IX1-NN*IP14*JFM
       IXX=IX1/135
       MTP=MNP+IRV*KIAA
       MTP=MTP/MAA/NSCB
       IX2=IX2+MTP

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST
IZZ2= IIX*ICS(IIXA) - IY*ICS(IYB) / I3
C AGC FOR BIT SYNC LOOP
IZZ2= I3
563 IZZ2= I3
564 IF (IZZ2.GE.0) 563, 564
412 LS(I)=IZZ2
413 IXX= IXX+IZZ2
IF (ICOUNI.GE.KH) 612, 100
612 IF (IDUM.LT.ISR) G25, 614
614 IGBIT=0
614 IGBIT=0
615 IF (IXX.GT.0) 615, 616
616 IXX= 0
625 IF (I.LE.KI) 626, 627
626 IA= IA+ IXX
627 IF (I.GT.KI.AND.I.LE.KJ) 628, 629
628 IB= IB+ IXX
629 IF (I.GT.KJ.AND.I.LE.KK) 630, 631
630 IG= IG+ IXX
631 IF (I.LT.KKI) 632, 633
632 IAZ= 1
633 IF (IAZ.EQ.1) 100, 634
634 L1=(L1+1)/I33
L2=(L2+1)/I33
IEI=(L1*L1-L2*L2)/ITP
IE= IE+2*IEI
IF (IEI.GE.IF) L= J, NTHAT= J, NTHAT+1, RHR
JTHAT= NTHAT+ISIGN(IFS2, NTHAT)
L = J, NTHAT = L, 18F
JTHAT = L, 18F
IF (L.IEQ.NTHAT) 700, 701
700 NTHAT = NTHAT+ISIGN(IFS2, NTHAT)
NTHAM = JTHAT
NTHAM = NTHAT
701 IF (IABS(IR), LT, 18F) 635, 637
639 IGN= 1
GO TO 100
637 IF (IABS(IR).GT.KI) 14, 15
14 IR= ISIGN(KI, IR)
15 IDUM= IDUM+ IR
1 IF (IDUM.GE.ISK) 1, 100
1 IDUM= IDUM+ ISR
10 IGBIT= 0
10 IF (ITX.GT.0) 2, 3
2 IGBIT= 1
3 IXX= 0
10 CONTINUE
C SSwitch(1) OFF FOR DEMODULATED BITS AND INPUT SIGNAL
C SSwitch(1) ON AND SSwitch(2) OFF FOR DEMODULATED BITS AND COSTAS LOOP PHASE
C SSwitch(1) ON AND SSwitch(2) ON FOR CORRELATOR INTEGRAL AND BIT SYNC
C Error Signal
GO TO (400,401) SWITCHF(1)

400 GO TO (403,404) SWITCHF(2)

403 NOUT(1)=((11X+13*ISR)/13)*15
NOUT(2)=40*IR+600
GO TO 402

404 NOUT(1)=10BIT*1000
NOUT(2)=(1XX+1F14)/4
GO TO 402

401 NOUT(1)=10BIT*1000
NOUT(2)=INT(1)+512

402 CONTINUE
CALL DISTWO(NOUT)
GO TO 101
END

V-11
APPENDIX VI

LISTING OF DIGITAL RECEIVER WITH SAMPLE
STORAGE BIT SYNCHRONIZATION (SPLIT-PHASE DATA)
AN = 4 * Bn * BN / (AMP * AMP)
DA = H * BN
GA = H * AN

740 IF (.011 * LE.BNA, AND .041 * GT.BNA) 740, 741
    ND = UA IG
    NNG = GA * 16 * 10
    NIX = 10 * I3
    NIX = 10 * I3
    NLSCAL = 13 / 10
    NNSCA = 1
    IF (SNR.LT. 5.00) 742, 741

742 I3 = 100

743 IF (.013 * LE.BNA, AND .037 * GT.BNA) 743, 744
    ND = UA * IG
    NNG = GA * 16 * 10
    NIX = I3
    NIX = 10 * I3
    NLSCAL = 13
    NNSCA = 10
    IF (SNR.LT. 5.00) 745, 744

745 I3 = 100

746 IF (.037 * LE.BNA, AND .118 * GT.BNA) 746, 747
    ND = UA * 15
    NNG = GA * 16
    NIX = I3
    NIX = 10 * I3
    NLSCAL = 13
    NNSCA = 10
    IF (SNR.LT. 5.00) 748, 747

748 I3 = 100

749 IF (.113 * LE.BNA, AND .187 * GT.BNA) 749, 750
    ND = UA * 15
    NNG = GA * 15
    NIX = I3
    NIX = 10 * I3
    NLSCAL = 13
    NNSCA = 10
    IF (SNR.LT. 5.00) 751, 750

751 I3 = 100

752 IF (.187 * LE.BNA, AND .375 * GT.BNA) 752, 753
    ND = UA * (15 / 10)
    NNG = GA * 15
    NIX = I3 / 10
    NIX = 13 / 10
    NLSCAL = 13 / 10
    NNSCA = 10
    IF (SNR.LT. 5.00) 754, 753

754 I3 = 100

755 IF (.375 * LE.BNA, AND .118 * GT.BNA) 755, 756
    ND = UA * (15 / 10)
    NNG = GA * (15 / 10)
    NIX = I3 / 10
    NIX = 13 / 10
    NLSCAL = 13 / 10
    NNSCA = 100
    IF (SNR.LT. 5.00) 757, 756
DIGITAL RECEIVER STUDY AND IMPLEMENTATION

IF (1.13 .LE. BNA AND .1.87 .GT. BNA) 753,759
NNG=DA*15/10
NIX=100
NIXX=1
NLSCAL=13*13/10
NSCA=13/10
IF (SNR.LT.5.00) 760,759
I38=100
I39=10
IF (1.87 .LE. BNA .AND. 3.75 .GT. BNA) 761,762
NNG=DA*13
NIX=10
NIXX=1
NLSCAL=13*13/10
NSCA=13
IF (SNR.LT.5.00) 763,762
I38=100
I39=10
IF (3.75 .LE. BNA .AND. 11.8 .GT. BNA) 764,765
NNG=DA*10
NIX=13/10
NIXX=1
NLSCAL=13*10
NSCA=100
I39=100
IF (SNR.LT.5.00) 766,765
I38=100
I39=10
CONTINUE
WRITE (6,500)
FORMAT (41X,38WHMCDONNELL DOUGLAS DIGITAL PSK RECEIVER),
WRITE (6,501)
FORMAT (41X,11DHBIT RATE IS,F10.5,2X,15MBS PER SECOND)
WRITE (6,502)
FORMAT (41X,14HSAMPLE RATE IS,15,2X,15MSAMPLES PER BIT)
WRITE (6,503)
FORMAT (41X,32HACQUISITION BANDWIDTH PARAMETERS)
WRITE (6,504)
FORMAT (41X,32HCOSTAS LOOP NOISE BANDWIDTH IS,F10.5,2X,2HZ)
WRITE (6,505)
FORMAT (41X,32HSYNCHRONIZATION LOOP NOISE BANDWIDTH IS,
F10.5,2X,2HZ)
WRITE (6,506)
FORMAT (41X,29HTACKING BANDWIDTH PARAMETERS)
WRITE (6,507)
WRITE (6,508)
MCT=0
IJD=0
IJD+1
MCT+1
MCT=0
MCT=0
MFLAG=0
IF (.0116 .LE. BWN3DB .AND. .0187 .GT. BWN3DB) 710,711
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>710</td>
<td>MD=0*I6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G<em>I6</em>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIX=10*I3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIXX=10*I3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LSCALE=13/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSCA=1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF (SNR, LT.5.00) 82,711</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I36=100</td>
<td></td>
</tr>
<tr>
<td>711</td>
<td>IF (.317, LE, BWN3DB, AND, .375, GT, BWN3DB) /12,713</td>
<td></td>
</tr>
<tr>
<td>712</td>
<td>MD=0*I5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G<em>I6</em>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIX=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIXX=10*I3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LSCALE=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSCA=10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF (SNR, LT.5.00) 83,713</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I36=100</td>
<td></td>
</tr>
<tr>
<td>713</td>
<td>IF (.3375, LE, BWN3DB, AND, .118, GT, BWN3DB) 714,715</td>
<td></td>
</tr>
<tr>
<td>714</td>
<td>MD=0*I5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G*I6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIX=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIXX=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LSCALE=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSCA=10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF (SNR, LT.5.00) 84,715</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I36=100</td>
<td></td>
</tr>
<tr>
<td>715</td>
<td>IF (.118, LE, BWN3DB, AND, .187, GT, BWN3DB) 719,717</td>
<td></td>
</tr>
<tr>
<td>716</td>
<td>MD=0*I5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G*I5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIX=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIXX=13/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LSCALE=13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSCA=10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF (SNR, LT.5.00) 85,717</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I36=100</td>
<td></td>
</tr>
<tr>
<td>717</td>
<td>IF (.187, LE, BWN3DB, AND, .375, GT, BWN3DB) 713,719</td>
<td></td>
</tr>
<tr>
<td>718</td>
<td>MD=0*(I5/10)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G*I5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIX=13/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIXX=13/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LSCALE=I3*10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSCA=100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF (SNR, LT.5.00) 86,719</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I36=100</td>
<td></td>
</tr>
<tr>
<td>719</td>
<td>IF (.375, LE, BWN3DB, AND, .118, GT, BWN3DB) 720,721</td>
<td></td>
</tr>
<tr>
<td>720</td>
<td>MD=0*(I5/10)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G*I5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIX=13/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KIXX=13/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LSCALE=I3*10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSCA=100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF (SNR, LT.5.00) 87,721</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I36=100</td>
<td></td>
</tr>
<tr>
<td>721</td>
<td>IF (.118, LE, BWN3DB, AND, .187, GT, BWN3DB) 722,723</td>
<td></td>
</tr>
<tr>
<td>722</td>
<td>MD=0*(I5/10)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MG=G*I3</td>
<td></td>
</tr>
</tbody>
</table>
KIAA=KIXX
LSCAL=LSCAL
NSCB=NSCA

IF B=LSF
18F2=LSF2
IC=LG
I3=I36
I35=I17
IF (KUX.EQ.SSUMTHF(I)) 790,792
792 IX1=IX1/LIX
IX2=IX2/LIX
JFL=JFL/LIX
JFM=JFM/LIX
JFN=JFN/LIX
GO TO 790
799 MT=NU
MR=NNG
KIA=NIX
KIAA=NIXX
LSCAL=LNCAL
NSCB=NSCA
I8F=ISF
I8F2=ISF2
IC=NG
I3=I38
I35=I39
IF (KUX.EQ.SSUMTHF(I)) 790,793

793 IX1=IX1*LIX
IX2=IX2*LIX
JFL=JFL*LIX
JFM=JFM*LIX
JFN=JFN*LIX
790 KUX=SSUMTHF(I)
IX=(ISQ*INT(1))72048
IY=(ISU*INT(2))/2048
C DC BIAS REMOVER
IAX=IAX/IAX/IAX/IAX/1/IX/INTA
IAY=IAY/IAY/IAY/IAY/IAY/IAY
IY=IY/IAY/IAY/IAY/IAY/IAY
IUN=IUN+1
ICOUNT=ICOUNT+1
ICUM=ICUM+1
609 IF (IAX-EQ.1) 8,609
610 IF (ICOUNT.EQ.KH) 7,613
7 IA=0
IC=0
IF (IR.EQ.0) 8,305
305 IF (IR.LT.0) 303,304
303 KHJ=KJI+IR
KHJJ=KHJ+KII
DO 300 IJK=KHJ,KHJJ
300 IA=IA+LS(IJK)
KK2=KK1+1K
LSS=KI

DO 301 IJK=KK2,KK

DO 302 IJK=KA,KK
301 LSS=LSS+1
I=K1-IR
GO TO 8
302 I=LA+L(IJK)
306 I=KI-IR
GO TO 8
78 IZN=0
IC=IC
IA=IC
IC=0
I=KI
8 IAZ=0
I=I+1
ICOUNT=KH

C COMMON TERMS FOR X1ANJ X2
C FIRST TERM
613 MAP=IX*2-IY*2/4/I3
MBP=ICS(IAX)*ICS(IAX)/I3
MP=MCP*MIF/I33
MPF=ICS(IAX)/I3
MIF=ICS(IAX)*ICS(IAX)/I3
MIF=MIF+MIF

C COMBINE MRP AND MCP TO USE IN BOTH EQUATIONS
MIRV=MCP+MIF
MF=IX*2-MIF/I34

C AGC FOR COSTAS LOOP
MAA=IX*2+IY*2/LSCAN
IF (MAA.EQ.0) 166,167
166 MAA=1
167 MIF=H*H/M/MAA
IX1=IX1+I*M+F*H
L=IX1/JFL
IX1=IX1+I*JFL
IF (IX1.JL.0) 202,203
202 IX1=IX1+I*JFL
203 IF (IX1.LJ=JM) 204,205
204 IXA=IP12
IXB=IP14
GO TO 206
205 IXA=IXA/I35
IXB=IXA+IP14
IF (IXB.EQ.IP12) 206,207
207 L=IXB/IP12
IX=IXB+L*IP12
206 MNN=IX1/IP1+SIGN(JFM,IX1)
MNN=MNN/JFM
IX1=IX1+MN*IP1+JFM
IX=IX1/135
JTHAT=L*IBF
ICR=JTHAT-NTHAM
NTHAM=JTHAT

700 IF (ICR,LT,100) 700,701
NTHAM=JTHAT-ISIGN(70000000,0,NTHAM)

701 IF (ICR,LT,100) 635,637
635 ICN=1
GO TO 100

637 IR=ICR*IBF
IF (ICR,GT,KI) 14,15
14 IR=SIGN(KI,IR)
15 IUO=IUO+IR
IUO=IUO+IR
IF (IUO,*GE,ISR) 1,100
1 IUO=IUO+ISR
IUO=IUO
IUO=IUO
GO TO 401

2 IOBIT=1
3 IF (IUO,*GE,K22) 35,100
35 IUO=IUO-K22
IOBIT=1
IF ((IUO=ITX),LE,J) 845,846

845 IOBIT=0
846 MAS=MAS+1
IF (100,NE,IOBIT) 840,841

840 MAR=MAR+1
841 IF (MAS,*GE,IV) 824,843
628 MAS=0
IF (MAR,*GE,MV) 800,830

800 MAR=0
GO TO 843

800 IUO=IUO+ISR
MAR=0
843 ITX=0
100 CONTINUE
C SSMTCHF(1) OFF FOR DEMODULATED BITS AND COSTAS LOOP PHASE
C SSMTCHF(1) ON AND SSMTCHF(2) OFF FOR CODED SPLT-PHASE AND INPUT SIGNAL
C SSMTCHF(1) ON AND SSMTCHF(2) ON FOR DEMODULATED BITS AND BIT SYNC ERROR SIGNAL
GO TO 10 (40,40) SSMTCHF(1)
404 GC 10 (40,40) SSMTCHF(2)
406 NUUT(1)=IOBIT*1000
NUUT(2)=40*IR+600
GO TO 408

407 NUUT(1)=IOBIT*1000
NUUT(2)=INT2+512
GO TO 408
408 CONTINUE
CALL DISW0(NOUT)
GO TO 101
END
FINIS
APPENDIX VII

LISTING OF BASEBAND DIGITAL RECEIVER
C  MCDONNEEL DOUGLAS DAS& BANDU DIGITAL PSK RECEIVER
DIMENSION LS(50),NOUT(2),INT(22)
NPTS=4
DO 9 INDX=1,50
9 LS(INDX)=0
C SNR IS SIGNAL-TO-NOISE RATIO
C NTAU IS NUMBER OF BIT PERIODS IN LFF TIME CONSTANT IN DC DIAS REMOVER
READ (5,119) SNR,NTAU
119 FORMAT (F20.6,I10)
121 IF (SNR.LT.5.00) 122,123
122 IEXP=2
GO TO 124
123 IEXP=3
124 IEXU=IEXP+2
IEXR=IEXP+3
I3=10**IEXP
I5=10**IEXQ
I6=10**IEXH
IJ=Ij/10
P1=3.14159
PI2=2*PI
IP1=3142
IP1C=2*IP1
IP14=IP1/2
C N IS THE NUMBER OF BIT PERIODS IN TIME CONSTANT OF THE BIT SYNC FILTER
C FOR ACQUISITION
C ISR IS THE NUMBER OF SAMPLES PER BIT
C BR IS THE NUMBER OF BITS PER SECOND
C SP IS PEAK SIGNAL VOLTAGE
READ (5,102) N,ISR,DR,SP
102 FORMAT (2110,F20.6)
ISF=(10.**I3)/SP
LSF=100*ISF
LSF=LSF/2
LG=4*N
KH=ISR/1+1
KI=ISR/2
KJ=2*KI
KK=3*KI
KHI=KI+1
KII=KI+1
KJ1=KJ+1
KK1=KK+1
I=0
ITP=4*ISR
H=1./((ISR*BR)
AMP=1.0
ZETA=SQR(T(2.)/T.0)
T=1./BR
IAI=0
IAY=0
IAY=10000
411IALPHA=NTAU*ISR
IFIV=IAI/ISR
C BITBW IS THE NOISE BANDWIDTH FOR THE BIT SYNCHRONIZATION LOOP
BITBW=(2*ZETA+1./I2.*ZETA))/(4.*ZETA*N*T)
C NB IS NUMBER OF BIT PERIODS IN TIME CONSTANT OF BIT SYNC FILTER
C FOR TRACKING
READ (5,103) NB
103 FORMAT (110,F20.6)
NSF=100*NB**2
NG=4*NB
BITdx=(2.*ZETA+1./((2.*ZETA**2)))/(4.*ZETA*NB**2)
BH=4.*BWA/(2.*ZETA+1./((2.*ZETA**2))
WRITE (5,500)
500 FORMAT (4X,38MCDOUGNELL DOUGLAS DIGITAL PSK RECEIVER)
WRITE (6,503) BR
503 FORMAT (142X,11BIT RATE IS,F10.5,2X,15BIT PER SECOND)
WRITE (6,505) ISR
505 FORMAT (142X,14SAMPLE RATE IS,152X,15SAMPLES PER BIT)
WRITE (6,506)
506 FORMAT (142X,32ACQUISITION BANDWIDTH PARAMETERS)
WRITE (6,502) BITdx
502 FORMAT (142X,43BIT SYNCHRONIZATION LOOP NOISE BANDWIDTH IS,)
WRITE (6,507)
507 FORMAT (142X,29TRACKING BANDWIDTH PARAMETERS)
WRITE (6,502) BITdx
MC=0
LDUN=0
101 CALL INODATA(INT,NPTS)
C SSWCHF(3) ON FOR TRACKING
C SSWCHF(3) OFF FOR ACQUISITION
GO TO (789,788) SSWCHF(3)
788 ISF=LSF
ISF2=LSF2
IG=LG
GO TO 790
789 ISF=NSF
ISF2=NSF2
IG=NG
790 CONTINUE:
IX=(4SQ*INT(1))/2048
IY=(4SQ*INT(2))/2048
C DC BIAS R:MOVE
IAX=IAX+IAX*IALPHA+IFIV*IIX/NTAU
IAY=IAY+IAY*IALPHA+IFIV*IY/NTAU
IX=IX+IAX/IAX
IY=IY+IAY/IAY
ICOUNT=ICOUNT+1
IDUM=IDUM+1

IF (IAZ.EQ.0) 8,609
609 IF (ICN.EQ.1) 78,510
610 IF (ICOUNT.GE.KH) 7,613

7 IA=0
IB=0
IC=0

IF (IR.EQ.0) 0,303
303 IF (IR.LT.0) 303,304
304 KHA=KHJ+IKJ
305 IA=IA+LS(IJK)
306 IA=IB+LS(IJK)
307 IA=IA+LS(IJK)
308 IA=IA+LS(IJK)
309 IA=IA+LS(IJK)
310 IA=IA+LS(IJK)
311 IA=IA+LS(IJK)
312 IA=IA+LS(IJK)
313 IA=IA+LS(IJK)
314 IA=IA+LS(IJK)
315 IA=IA+LS(IJK)
316 IA=IA+LS(IJK)

78 ICN=0
IA=IC
IB=0
IC=0
I=KI

8 IAZ=0
I=I+1
613 CONTINUE
ICOUNT=KH
IY=0

C AGC FOR BIT SYNC
IZAZ=-I3
150 IZAZ=I3
151 IZAZ=I3

IF (ICOUNT.GE.KH) 512,413
412 LS(I)=IZAZ
413 IY=IY+IZAZ

IF (ICOUNT.EQ.KH) 612,100
612 IF (IDUM.LT.ISR) 625,614
614 I0BIT=0
615 I0BIT=1
616 IY=0
IDUM=0

625 IF (I.I.EQ.KI) 628,629
626 IA=IA+IZAZ
627 IF (I.GT.KI.AND.I.LE.KJ) 528,629
628 IE=I+1A2
629 IF (I.GT.KJ. AND. I.LE.KK) 630, 631
630 IC=IC+1A2
631 IF (I.LT.KK) 632, 533
632 IAZ=1
633 IF (IAZ.EQ.1) 100, 634
634 L1=-(IA+IB)/133
L2=(IB+IC)/133
IE=(L1*L1-L2*L2)/IIP
IE=IE+2*III
IF=IE+16*IE
JTHAT=JTHAT*ISIGN(ISF2,NTHAT)
L=JTHAT/ISF
JTHAT=JTHAT-NTHAM
NTHAM=JTHAT
IF (LABS(NTHAT),GE,7000000) 700, 701
700 NTHAM=NTHAM-NTHAM-lSIGN(7000000,NTHAT)
NTHAM=NTHAM-ISIGN(7000000,NTHAM)
701 IF (LABS(1CR),LT,ISF) 635, 637
635 ICN=1
GO TO 100
637 IF =ICR/ISF
IF (LABS(IR),GT,KI) 14, 15
14 IR=ISIGN(KI,IR)
15 IOM=IODM-IR
2 IF (IOM.GE.ISR) 1, 100
1 IOM=IODM-ISR
100 CONTINUE
C SSMTCFF(1) OFF FOR DEMODULATED BITS AND INPUT SIGNAL
C SSMTCFF(1) ON AND SSMTCFF(2) OFF FOR DEMODULATED BITS AND BIT SYNC
C ERROR SIGNAL
C SSMTCFF(1) ON AND SSMTCFF(2) ON FOR DEMODULATED BITS AND CORRELATOR INTEGRAL
GO TO (-04, 05) SSMTCFF(1)
404 GO TO (+06, 07) SSMTCFF(2)
406 NOUT(1)=IOBIT*1000
NOUT(2)=((11X+13*ISR)/13)*15
GO TO 408
407 NOUT(1)=IOBIT*1000
NOUT(2)=40*IR+600
GO TO 408
405 NOUT(1)=IOBIT*1000
NOUT(2)=INT(2)*512
408 CONTINUE
C CALL UISTH0(NOUT)
GO TO 101
END
FINISH
APPENDIX VIII

LISTING OF DIGITAL RECEIVER WITH SWEPT FREQUENCY ACQUISITION
PROGRAM BIT
COMMON ICS
C MCDONNELL DOUGLAS DIGITAL PSK RECEIVER
C RECEIVER HAS DC BIAS REMOVER, AGC, SAMPLE STORAGE, BIT SYNC, AND SWEPT
C FREQUENCY ACQUISITION
DIMENSION ICS(6285), LS(50), NOUT(2), INT(22)
NPTS=4
NO 9 INDX=1, 50
LS(INDX)=0
C SFN IS SIGNAL TO NOISE RATIO IN DB
C NTAU IS TIME CONSTANT OF LPF IN DC BIAS REMOVER TO BIT PERIODS
READ (5,119) SNR, NTAU
119 FORMAT (F20.6,110)
121 IF (SNR.LT.5.00) 122,123
122 IEXP=2
123 IEXP=3
124 IE XO=IE XP+2
IE XR=IE XP+3
N3=10**IE XP
N5=10**IE XO
N6=10**IE XR
N33=N3/10
N38=N33
N39=N3
N36=N33
N37=N3
PI=3.14159
PI2=2*PI
PI3=3142
PI4=2*PI
C N IS TIME CONSTANT OF BIT SYNC FILTER IN BIT PERIODS
C ISR IS THE NUMBER OF SAMPLES PER BIT
C BR IS THE NUMBER OF BITS PER SECOND
C BWNO3DB IS NOISE BANDWIDTH IN HZ FOR COSTAS LOOP
C SP IS PEAK SIGNAL VOLTAGE
READ (5,102) N, ISR, BR, BWNO3DB, SP
102 FORMAT (2I10,3F20.6)
ISQ=100.*N**2
LSF=100.*N**2/SP
LSF2=LSF/2
L G=4*N
KH=ISR/4+1
KI=ISR/2
KJ=2*KI
KK=3*KI
KMI=KI+1
KII=KI-1
KJ1=KJ+1
KJI=KK+1
I=0
IIFP=4*ISR
H=1./(ISR*BR)
AMP=1.0
ZETA=SQR(2.1/2.0)
KIX=10*I3
KIXX=10*I3
LSCAL=I3/10
NSCA=1
IF (SNR.LT.5.00) 82,711
I36=100
711 IF (.0187.LE.BWN30B.AND..0375.GT.BWN30B) 712,713
712 MD=0*I5
MG=G*I6*10
KIX=13
KIXX=10*I3
LSCAL=I3
NSCA=10
IF (SNR.LT.5.00) 83,713
I36=100
713 IF (.0375.LE.BWN30B.AND..118.GT.BWN30B) 714,715
714 MD=0*I5
MG=G*I6
KIX=13
KIXX=I3
LSCAL=I3
NSCA=10
IF (SNR.LT.5.00) 84,715
I36=100
715 IF (.118.LE.BWN30B.AND..197.GT.BWN30B) 716,717
716 MD=0*I5
MG=G*I5
KIX=13
KIXX=I3/10
LSCAL=I3
NSCA=10
IF (SNR.LT.5.00) 85,717
I36=100
717 IF (.187.LE.BWN30B.AND..375.GT.BWN30B) 718,719
718 MD=0*(15/10)
MG=G*I5
KIX=13/10
KIXX=I3/10
LSCAL=I3*10
NSCA=100
IF (SNR.LT.5.00) 86,719
I36=100
719 IF (.375.LE.BWN30B.AND..197.GT.3WN3DB) 720,721
720 MD=0*(15/10)
MG=G*(15/10)
KIX=I3/10
KIXX=I3/100
LSCAL=I3*10
NSCA=100
IF (SNR.LT.5.00) 87,721
I36=100
721 IF (1.18.LE.BWN30B.AND..1.67.GT.8WN3DB) 722,723
722 MD=0*(15/10)
MG=G*I3
KIX=100
KIXX=1
LSCALE=I3*I3/100
NSCA=I3/10
IF (SNR.LE.5.00) 730,723

730 JFL=JFL/10
    JFM=JFM/10
    JFN=JFN/10
    I36=100
    I37=10

723 IF (1.87.LE.BWN1DB.AND.3.75.GT.BWN3DB) 724,725

724 MD=0*I3
    MG=G*I3
    KIX=10
    KIXX=1
    LSCALE=I3*I3/10
    NSCA=I3
    IF (SNR.LE.5.00) 731,725

731 JFL=JFL/10
    JFM=JFM/10
    JFN=JFN/10
    I36=100
    I37=10

725 IF (3.75.LE.BWN1DB.AND.11.8.GT.BWN3DB) 726,727

726 MD=0*I3
    MG=G*I3
    KIX=I3/10
    KIXX=1
    LSCALE=I3*I3/10
    NSCA=100
    JFL=JFL/10
    JFM=JFM/10
    JFN=JFN/10
    I36=100
    I37=100

VIII

727 IX1=JFL
    IX1=1
    KOK=2

    MIAU=800
    KIAT=0
    KFLAG=0
    MENT=10
    IF (SNR.LE.5.00 .AND.BWN3DB.GE.1.10) 3000,3001

3000 MENT=1
3001 CONTINUE
    IF (SNR.GT.5.00 .AND.BWN3DB.LT.3.75) 3002,3003

3002 MENT=100
3003 CONTINUE

C SWEEP START IS -PI RAD/SEC
C DW IS ONE HALF THE SWEEP RATE IN RAD/SEC/SEC
C DW IS NOMINALLY SET AT 0.025 RAD/SEC/SEC
READ (5,119) DW
DIGITAL RECEIVER STUDY MDC E0648

JUNE 1972

C PERC SETS THRESHOLD ON ACQUISITION INDICATOR
C PERC HAS MAXIMUM VALUE OF 1.0 AND IS NOMINALLY SET AT 0.98

PERC=0.98
ITHI=(ISX*I13)*PERC
C ENTRY POINT FOR X AND Y IS 101
C X AND Y ARE QUADRATURE COMPONENTS
101 CALL INDATA(INTNFTS)
C TURN SSNCHF(3) ON AND THEN OFF TO MANUALLY RESET SWEEP
GO TO (393,399) SSNCHF(3)

398 KFLAG=0
IKX=JFL
IYA=IP12
IXB=IP14
IX2=0
IX3=0
ICAN=0
KIA0=0
399 CONTINUE
788 MY=0
MS=0

KIA=KIX
KIAA=KXX
LSCAN=LSCAL
NSCA=NSCA
IIF=LSF
IIF2=LSF2
IO=LG
I34=I36
I35=I37
IX=(ISQ*INT(1))/2048
IT=(ISQ*INT(2))/2048
C DC BIAS REMOVER
IAX=IAZ+IALPHA*IFIV*IX/NTAU
IAY=IAY+IALPHA*IFIV*IY/NTAU
IX=IX+IALpha
IY=IY+IYAA
ICOUNT=ICOUNT+1
IGN=IGN+1
IF (IAZ<0.1) 8,609
609 IF (IGN<0.1) 79,610
610 IF (ICOUNT-GE.KM) 7,613
7 IA=0
IB=0
IC=0
IF (IR.EQ.0) 8,305
305 IF (IR.LT.0) 303,104
303 KHJK=KJ1+IR
KHHJ=KHJK+KJ1
DO 300 IJK=KHJK,KHJ
300  IA=IA+LS(I,K)
    KK2=KK1+IR
    LSS=KHI
    DO 301 IJK=KK2,KK
        IA=IA+LS(I,JK)
        LSS=LS(IJK)
    301  I=I-IK
        GO TO 8

304  IF (IR,E,I) 306,307
307  KA=KJ1+IR
    DO 302 IJK=KA,KK
    302  IA=IA+LS(IJK)
    306  I=I-IK
        GO TO 8

78  IGN=0
    IA=IG
    IC=0
    I=KI
     8  IAI=0
        I=I-1
        ICOUNT=KH
        MAP=(IX**2-IY**2)*4/I3
        MNP=ICS(IK)*IGS(XJ)/I3
        MDP=MAP*MAP/I3
        MNP=(IX*Y)/2/I3
        MNP=ICS(IK)*IGS(XJ)/I3
        MNP=MAP*MAP
        MIRV=MIRV+MNP
        MF=IX2*MJ/I34

C AGC
FOR COSTS LOOP,
    MAA=(IX**2-IY**2)/LSCAN
    IF (MAA .LE. 0) 166,167

166  MAA=1
    167  MEP=MT*MIRV/MAA
        IX1=IX1+PF*MEP
        L=IX1/JFL
        IX1=IX1-L*JFL
        IF (IX1.LE.0) 202,203

202  IX1=IX1+JFL
C FREQUENCY SWEEP LOGIC
C SWEEP WILL AUTOMATICALLY RESET IF RECEIVER DOES NOT ACQUIRE
C SWEEP WILL AUTOMATICALLY START IF ACQUISITION INDICATOR FALLS BELOW
C THR=SHOLD AFTER ACQUISITION
203  IF (KFLAG .EQ. 0) 215,214

215  CONTINUE
    ICAN=ICAN+1
    IF (ICAN GT 100000) 1114,1115

1114  ICAN=0
        MANG=0
        NANG=0
        JHAS=0
        LANG=0
1115  CONTINUE
MANG=MANG+INTER
L=MANG/62832
MANG=MANG-L*62832

IF (MANG.LE.0) 2000,2001
2000 MANG=MANG+62832
2001 CONTINUE
NDEL=IPRO0+2*IPR00*ICAN
L=LANG/6233165
LANG=LANG-L*6283165
IF (LANG.LE.0) 2022,2023
2022 LANG=LANG/6283165
2023 CONTINUE
LANG=LANG+NDEL/100
NANG=LANG/100
L=NANG/62332
NANG=NLG-N*62832
IF (NANG.LE.0) 2010,2011
2010 NANG=NANG+62832
2011 CONTINUE
JPHAS=MANG+NANG
JPHAS=JPHAS*MENT

1199 CONTINUE
L=JPHAS/JFL
JPHAS=JPHAS-L*JFL
IF (JPHAS.LE.0) 220,221
220 JPHAS=JPHAS+JFL
221 CONTINUE
IF (KZAX.GE.ITH1) 212,214

212 IX2=-IPI+(IPRA*ICAN)/1000
KFL46=1
JPHAS=0
LANG=0
MANG=0
NANG=0

214 CONTINUE
C END FREQUENCY SWEEP LOGIC
IX3=IX4+JPHAS
L=IX3/JFL
IX3=IX3-L*JFL
IF (IX3.LE.0) 230,231
230 IX3=IX3+JFL
231 IF (IX3.LT.JFM) 204,205
204 IXA=IPI2
IXB=IPI4
GO TO 205
205 IXA=IX3/IX5
IXN=IXA+IPI4
IF (IXN.LE.IPI2) 206,207
207 L=IXB/IPI2
IXN=IXB-L*IPI2
NN=IX3/IPI+ISIGN(JFM,IX3)
NN=NN+JFM
IXX=IX3-NN*IPI*JFM
IXX=IXX/IIX
NN=JPHAS/IPI+ISIGN(JFM,JPHAS)
NN=NN+JFM
IF (IDUM.GE.ISR) 1,100
   1 IDUM=IDUM-ISR
   IOBIT=0
   IVARB=IABS(IY)
   IVARB=IVARB*1000
   KZAZ=KZA7-KZAZ*MTAU+IVARB/MTAU
   KZAX=KZAZ/1000
   IF (ITX,GT.0) 2,3
  2 IOBIT=1
   IY=0
  100 CONTINUE
C $SWITCHF(1)$ OFF FOR ACQUISITION INDICATOR AND PHASE OF FREQUENCY SWEET
C $SWITCHF(1)$ ON AND $SWITCHF(2)$ OFF FOR DEMOULATED BITS AND COSTAS LOOP PHASE
C $SWITCHF(1)$ ON AND $SWITCHF(2)$ ON FOR DEMOULATED BITS AND INPUT SIGNAL
  400 GO TO (401,404) $SWITCHF(2)$
  401 NOUT(1)=IOBIT*1000
   NOUT(2)=INT(1)+512
   GO TO 402
   402 NOUT(1)=IOBIT*1000
   NOUT(2)=(IY+IP4)/4
   GO TO 402
  404 NOUT(1)=(KZAX/I3)*40
   NOUT(2)=(IY+IP4)/4
   CONTINUE
C $DISTNU(HOUT)$
   GO TO 101
END
FINIS