DEBUGGING EMBEDDED COMPUTER PROGRAMS

Gilbert H. Kemp
General Dynamics/Western Data Systems Center
Pamona, California

The debugging of embedded digital computer programs is a function that can use a wide range of support tools from essentially none to sophisticated systems. Every embedded computer program must complete its debugging cycle using some system that will allow real time debugging.

A listing of many of the common items addressed during debugging is given. Several approaches to debugging are analyzed to evaluate how well they treat those items. Cost evaluations are also included in the comparison.

The approaches compared are: (1) no software support, (2) embedded computer augmented with additional software for debugging purposes, (3) microprocessor development systems, (4) an environment simulation and interpretive computer simulation combination run on a large scale computer, (5) an environment simulation on a hybrid computer coupled with the embedded computer, (6) an environment simulation on a midi-computer coupled with an emulation of the embedded computer, and (7) an environment simulation on a midi-computer coupled with a slightly modified embedded computer.

The results of the comparison indicates that the best collection of capabilities to cover the common items present in the debugging task occurs in the approach where a midi-computer handles the environment simulation with an emulation of some kind representing the embedded computer. This approach can be taken at a reasonable cost.

The case study chosen is an embedded computer in a tactical missile. Several choices of computer for the environment simulation are discussed as well as different approaches to the embedded computer emulator.

The selected choice for computer the environment simulation is a special-purpose computer designed to rapidly solve differential equations. The Applied Dynamics International AD-10 computer is an example of this type. This appears to be capable of solving a full 6 Degree of Freedom missile simulation in real time.

The proposed choice for emulating the embedded computer is to use a modified version of the embedded computer itself. It is called an "Extended" computer. The "extension" amounts to adding extra bits to the program memory. When an instruction is loaded for execution, an interrupt will occur if one of these extra bits is up. Software interrupt service routines will determine what debugging function is to occur, e.g., start a trace.

The conclusion is that the use of the AD-10 and "Extended" embedded computer will show a debugging cost savings approaching 44 percent over the current commonly used team of Interpretive Computer Simulation used in conjunction with a hybrid/embedded computer pair.

Reference

Glass, Robert L., "Real-Time: The 'Lost World' Of Software Debugging and Testing", Comm. ACM 23, 5 (May 1980), Pages 264-271.

DEBUGGING EMBEDDED COMPUTER PROGRAMS

FOR ANY TASK:

- o IMPROVED PRODUCTIVITY IS DESIRABLE
- o BETTER TOOLS CAN IMPROVE PRODUCTIVITY
- o THE TOOLS MUST BE COST EFFECTIVE

APPLYING THESE THOUGHTS TO THE TASK OF DEBUGGING EMBEDDED DIGITAL COMPUTER PROGRAMS WILL BE DISCUSSED.

POMONA DIVISION DESIGNED DIGITAL COMPUTERS

COMPUTER	YEAR DESIGNED	NUMBER OF INSTRUCT.	NUMBER OF GEN.PURPOSE REGISTERS	INTER-RUPTS	SHORTEST INSTRUCTION TIME(μSEC.)	RELATIVE TIME OF EXECUTION	TECH-NOLOGY	MICRO-CODED
A	1971	24	2	NONE	0.4	1.0	HARD WIRED	NO
B	1975	74	4	ONE	0.9	3.1	BIT SLICE-INTEL 3000	YES
C	1976	102	8(16)	VECTORED	0.4	1.2	BIT SLICE-AMD 2900	YES
D	1978	158	9(17)	VECTORED	0.3	0.8	BIT SLICE-AMD 2900	YES

ITEMS IN DEBUGGING

A. INPUT -- HARDWARE AND CONVERSION
B. OUTPUT -- HARDWARE AND CONVERSION
C. MATHEMATICAL CALCULATIONS
D. LOGIC DECISIONS
E. CHECK ALL PATHS OUT OF DECISIONS
F. OVERFLOWS
G. MAXIMUM USE OF PRECISION WITHOUT OVERFLOW
H. LACK OF SIGNIFICANCE
I. INITIALIZATION OF VARIABLES
J. TIMING OF THE PROGRAM
K. PROCESS SWITCHING

DEBUG TECHNIQUES

A. TRACING SELECTED SECTIONS OF THE PROGRAM AT APPROPRIATE TIMES

B. ANALOG RECORDING OF SELECTED VARIABLES OF THE PROGRAM

C. PRINTING SELECTED PROGRAM VARIABLES AT APPROPRIATE TIMES

D. STOPPING ON BREAKPOINTS AND EXAMINING CONTENTS OF REGISTERS AND MEMORY

COMPARISON OF DEBUG TECHNIQUES

TYPE OF SIMULATION	COST OF DEVELOPMENT OF SIMULATION (4)	COST OF USE OF SIMULATION (4)	CAPABILITIES (4) (VERSATILITY)	USEFULNESS (4)	TYPE OF INPUTS (5)	SPEED (7)	USEFULNESS IN TESTING (8)											TYPE OF OUTPUT (9)
							A	B	C	D	E	F	G	H	I	J	K	
NONE--MFR. SOFTWARE	N	N	S	S	L/S	RT	-	-	X	-			-	-				E
NONE--ADD DEBUG SOFTWARE	S	N	S	S	L/S	RT	-	-	X	-			-	-			-	E
MICROPROCESSOR DEVELOPMENT SYSTEM (MDS)	S	S-	S	S	L,R	RT	-	-	X	-			-	-		-		P,T,E (6)
SIMULATION/I.C.S. (1) --CLOSED LOOP	H	H	H	H-	L	S	-	-	X	X	X	X	X	X	X	X	X	P,T,A,E
HYBRID (3)--CLOSED LOOP	H	H	M (10)	M	L,R	RT	X	X	-	X	-						-	A,E
MIDI/EMULATION (2) --CLOSED LOOP	M	S	H	H	L,R(?)	RT(?)/S	X	X	X	X	X	X	X	X	X	X	X	P,T,A,E

NOTES

(1) THE FLIGHT COMPUTER IS SIMULATION ON A LARGE SCIENTIFIC COMPUTER. THIS IS AN INTERPRETIVE COMPUTER SIMULATION (I.C.S.). IT IS MATED TO A FORTRAN MISSILE SIMULATION, NORMALLY IS 5 DEGREE OF FREEDOM (D.O.F.) BUT MAY BE 6 D.O.F.
(2) A 32-BIT MIDI-COMPUTER, HOOKED TO AN EMULATION OF THE FLIGHT COMPUTER WHOSE PROGRAM IS UNDER TEST, RUNS THE MISSILE SIMULATION OF (1). HARDWARE RECEIVER SIMULATOR.
(3) A FULL 6 D.O.F. MISSILE SIMULATION PUT TOGETHER FOR THIS TESTING, PLUS DESIGN AND EVALUATION OF EQUATION IS USED. HARDWARE RECEIVER SIMULATION OFTEN USED.
(4) N = NONE, S = SMALL, M = MEDIUM, H = HIGH.
(5) L = SIMULATED LIVE, S = STATIC VALUES, R = SIMULATED INPUTS TO REAL HARDWARE.
(6) TIMING INVOLVES THE USE OF A LOGIC ANALYZER.
(7) RT = REAL TIME, S = $<$RT.
(8) LETTERS ARE FROM THE TYPES OF CODE LISTED IN TABLE 2. X INDICATES GOOD CAPABILITY, (-) INDICATES THE CAPABILITY IS NOT AS GOOD AS OTHER APPROACHES AND A BLANK INDICATES NO CAPABILITY.
(9) E = EXAMINE ONE VARIABLE AT A TIME, A = ANALOG TYPE OUTPUT, P = PRINTED OUTPUT, T = TRACING OUTPUT.
(10) WHEN USING HYBRID, SCALING OF VARIABLES IS NOT EASY TO CHECK. IF IT WERE, THEN WOULD RATE H.

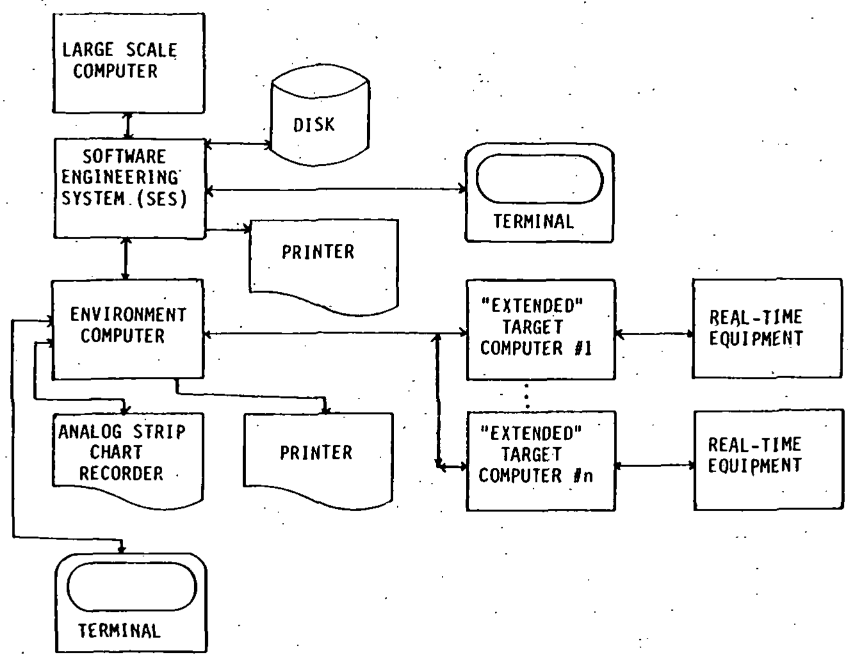
EMULATION APPROACHES

ENVIRONMENT COMPUTER	EMULATION COMPUTER
VAX	QM-1
VAX	"EXTENDED" TARGET
AD-10	"EXTENDED" TARGET

FUNCTIONS OF THE ADDED BITS

BITS	FUNCTION
0001X	IGNORE AN OVERFLOW ON THIS INSTRUCTION IF IT OCCURS, OTHERWISE PRINT AN ERROR MESSAGE IF AN OVERFLOW OCCURS.
0101X	CAUSE PROGRAM TO BECOME SYNCHRONIZED WITH ENVIRONMENT COMPUTER PROGRAM.
0100X	BREAKPOINT.
1000X	START A TRACE--IF TIME HAS REACHED A SPECIFIED VALUE. ALSO SET A FLIP-FLOP TO COMMAND AN INTERRUPT TO PRINTOUT TRACE DATA ON EACH SUCCEEDING INSTRUCTION.
1010X	STOP TRACING, I.E., RESET THE TRACE FLIP-FLOP.
1100X	START TIMING FROM POINT A TO POINT B IF RUN TIME HAS REACHED A SPECIFIED VALUE, I.E., READ THE CLOCK OF TARGET OR ENVIRONMENT COMPUTER.
1110X	STOP TIMING--READ THE CLOCK OF TARGET OR ENVIRONMENT COMPUTER AND PRINT THE DELTA TIME SINCE TIMING STARTED.

ENVIRONMENT COMPUTER/"EXTENDED" TARGET COMPUTER BLOCK DIAGRAM



OVERALL COMPARISON OF DEBUG TECHNIQUES

APPROACH	ADVANTAGES	DISADVANTAGES
ACTUAL TARGET COMPUTER.	LEAST EXPENSIVE RE HARDWARE.	NOT ENOUGH TOOLS. TOO PRIMITIVE.
MDS	INEXPENSIVE RE HARDWARE.	NOT ENOUGH TOOLS. TOO PRIMITIVE.
INTERPRETIVE COMPUTER SIMULATION (I.C.S.) ON A LARGE COMPUTER PLUS HYBRID PAIR--USED SEPARATELY.	GOOD SET OF TOOLS. HYBRID IS REAL-TIME. HYBRID USES REAL TARGET COMPUTER. CURRENT METHOD IN USE.	VERY EXPENSIVE RE HARDWARE AND OPERATING COSTS. I.C.S. TURN AROUND IS SLOW.
VAX FOR MISSILE SIMULATION PLUS QM-1 FOR TARGET EMULATION.	GOOD SET OF TOOLS. EMULATE NON-EXISTENT TARGET COMPUTER. LESS EXPENSIVE THAN I.C.S. PLUS HYBRID.	VERY SLOW. MULTI-CPU TARGET EVEN SLOWER.
VAX FOR MISSILE SIMULATION PLUS "EXTENDED" TARGET COMPUTER.	GOOD SET OF TOOLS. USES REAL TARGET COMPUTER. ALLOWS MULTI-CPU TARGET. LESS EXPENSIVE THAN VAX PLUS QM-1.	NOT REAL-TIME. REQUIRES VERIFICATION OF "EXTENDED" TARGET COMPUTER CONCEPT.
AD10 FOR MISSILE SIMULATION PLUS "EXTENDED" TARGET COMPUTER.	GOOD SET OF TOOLS. USES REAL TARGET COMPUTER. ALLOWS MULTI-CPU TARGET. REAL-TIME! LESS EXPENSIVE THAN VAX PLUS "EXTENDED" TARGET. BEST OVERALL APPROACH.	REQUIRES VERIFICATION OF AD10 SPEED. REQUIRES VERIFICATION OF "EXTENDED" TARGET COMPUTER CONCEPT.

EMULATION METHODS SUMMARY

ITEM	APPROACH		
	VAX/QM-1	VAX/EXT.	AD10/EXT.
FULL DEBUGGING TOOL SET	YES	YES	YES
MULTIPLE CPU'S	YES-SLOW	YES	YES
"ACTUAL" HARDWARE	NO	ALMOST	ALMOST
REAL-TIME	NO	NO	YES
FULLY REPLACE HYBRID?	NO	NO	YES
EASE OF ENVIRONMENT COMPUTER PROGRAMMING	EASY	EASY	HARDER
"RUN" A NON-EXISTENT COMPUTER	YES	NO	NO
NET SAVINGS OVER PRESENT METHOD (HYBRID/ICS) OF FEATURES THAT DIFFER BETWEEN THE FOUR APPROACHES (2 MAJOR WEAPONS PROGRAMS)	17%	32%	44%