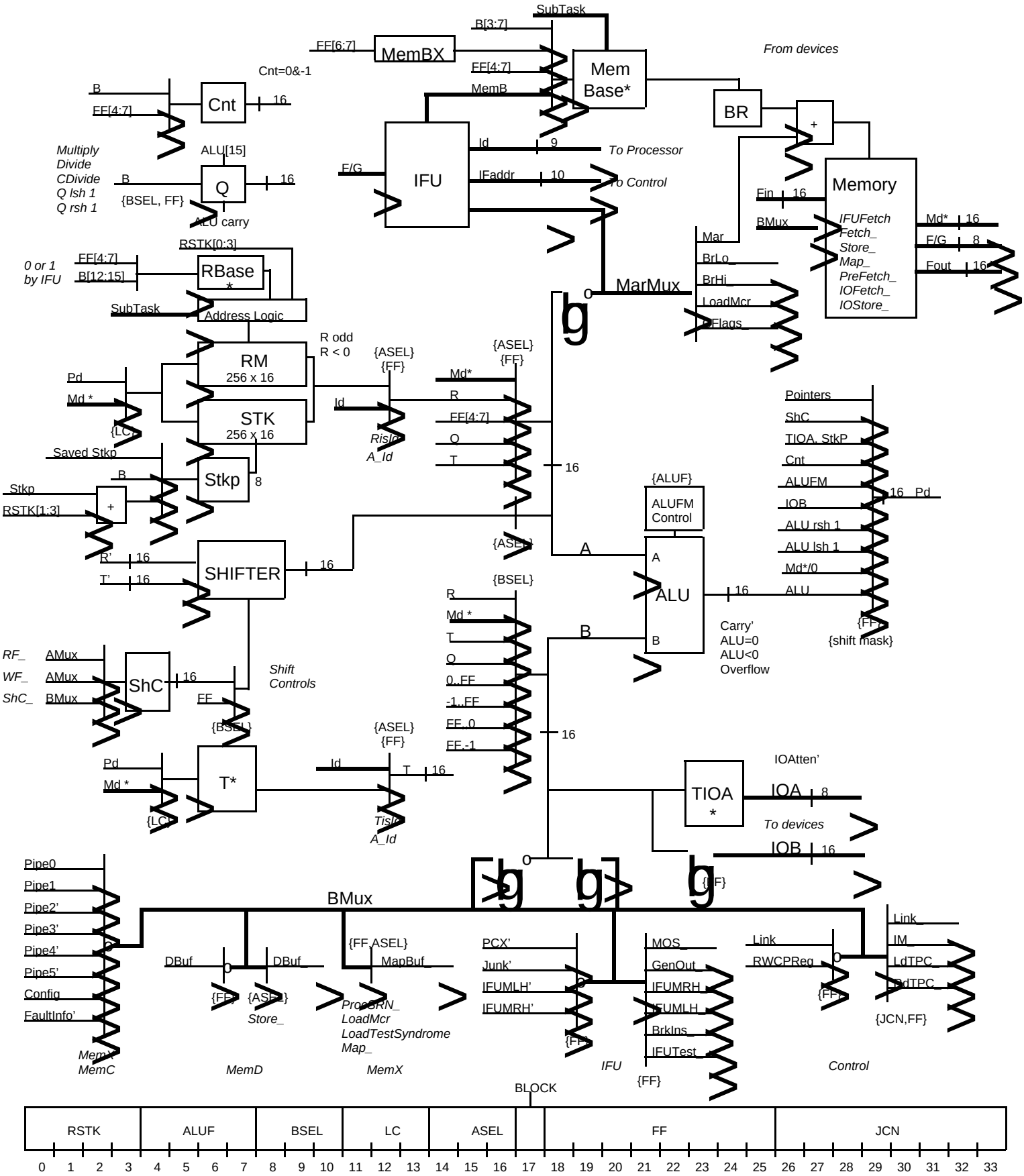
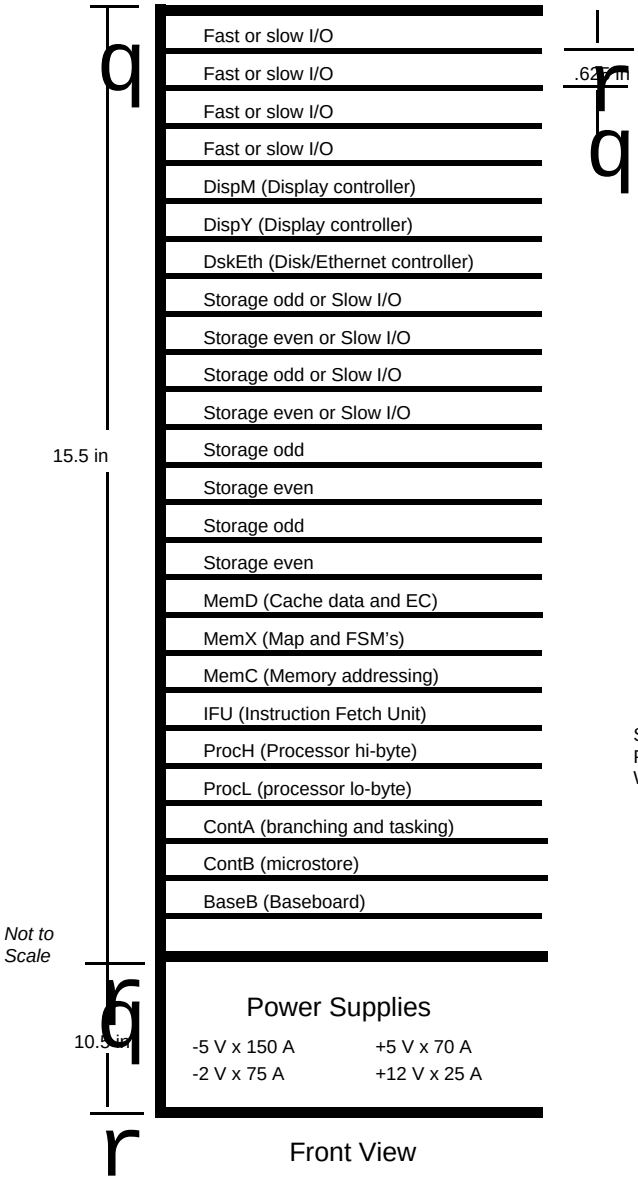


* Task-Specific

{xxx} Source of Control

Figure 1
Dorado
Programmer's View

D1ProgView.sil
8/31/79



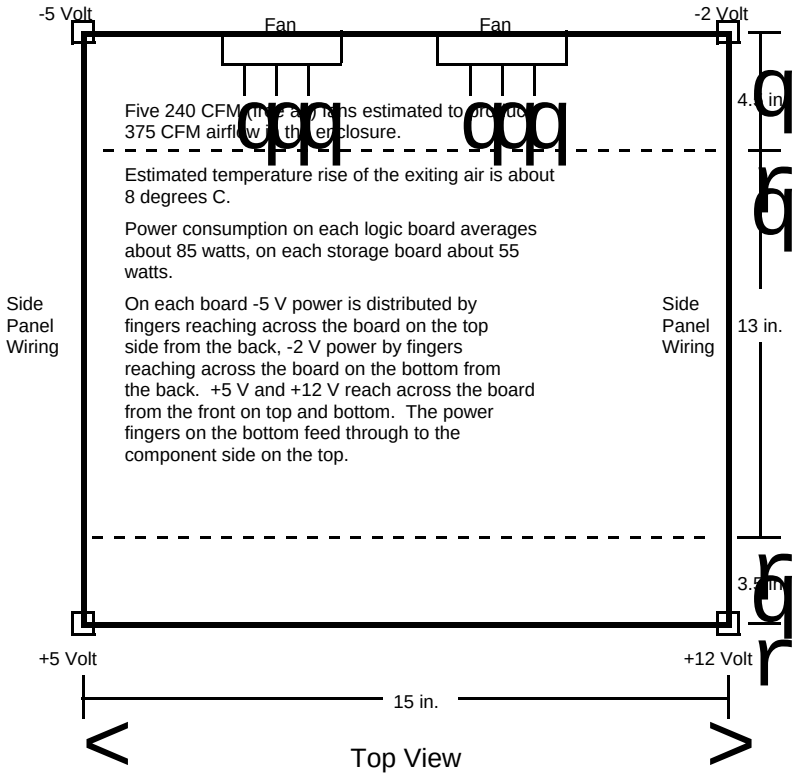
The +5 V supply and one fan are controlled by a switch; -5 V, +12 V, and -2 V supplies, the other four fans, and the disk logic and spindle power are controlled by the baseboard microcomputer (or the controlling Alto).

BaseB and ContB boards are equipped with temperature sensors that are repetitively monitored by the baseboard microcomputer; most other boards have temperature sensors that can be monitored when the microprocessor is halted. In the event some temperature exceeds 60 degrees C, the microcomputer will shut down the three power supplies that it controls.

The microcomputer also monitors power supplies; when any voltage or current deviates from its allowed range, the microcomputer shuts off power to the three supplies that it controls.

The card cage shown here is beneath the Trident T80 disk, and both are inside an enclosure designed to reduce the noise level for an office environment. The total enclosure size is about 4 feet high x 4 feet deep x 2 feet wide (ugh).

The machine weighs between 500 and 600 lbs.



The 11 logic boards in production models will be multiwire; 2 to 8 storage boards and the two backpanels are printed circuits.

The following shows approximate component count:

11 Logic boards:		*8 Storage boards:*	
2315	ic's of random logic	1056	ic's of random logic
246	1kx1 ECL RAM's	1152	16Kx1 MOS RAM's
71	16x4 ECL RAM's		
24	256x4 ECL RAM's		
21	16Kx1 MOS RAM's		
1600	SIP's		

Each board can mount 24 x 12 or 288 16-pin DIPs. Normal MECL-10000 DIP's are connected to the ground plane and -5V supply. Logic nets are terminated through 100-ohm resistors at one or both ends to the -2V supply. The resistors are in low-profile SIPs that mount between the DIPs (144 8-pin SIPs per board).

The +5 V supply is used for TTL/ECL conversions and for TTL components. The MOS IC's on the memory storage boards and in the Map use the +12 V supply.

Figure 2
Card Cage

D1CardCage.sil
9/14/81

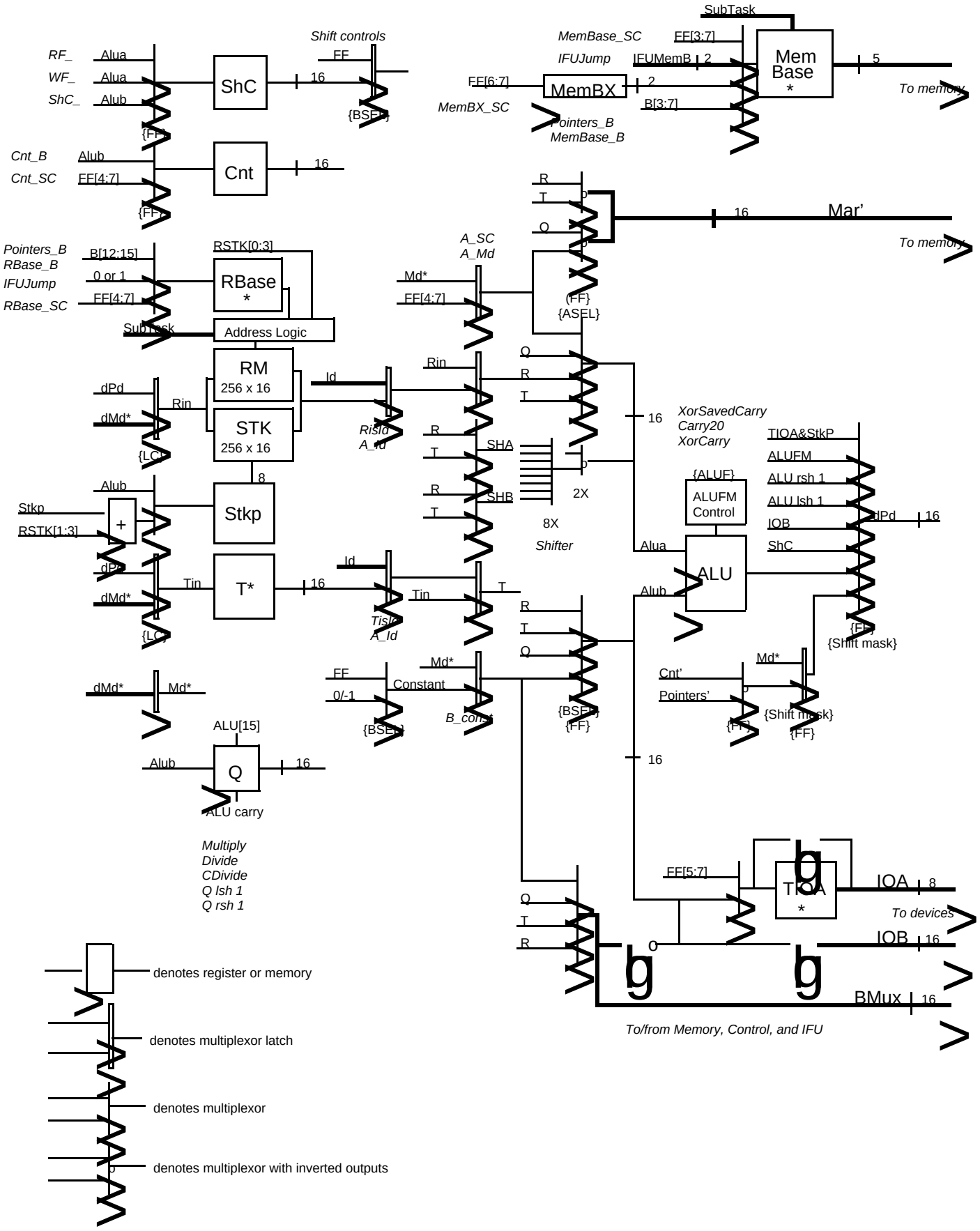
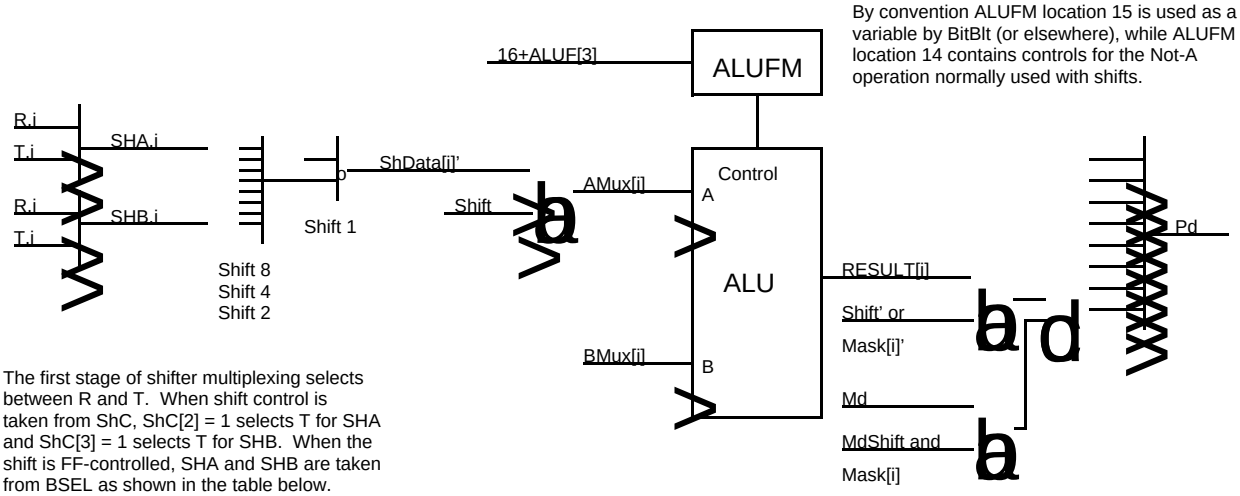


Figure 3
Processor
Hardware View

* Task-Specific



Shift Data Paths

Field	SHA	SHB	Shift Count	RMask	LMask
ShC bits:	2	3	4:7	8:11	12:15
RF_A	A[2]	A[3]	P+S+1	undefined	15-S
WF_A	A[2]	A[3]	16-P-S-1	16-P-S-1	P
ShC_B	B[2]	B[3]	B[4:7]	B[8:11]	B[12:15]
BSEL.0=1	BSEL.1	BSEL.2	FF[4:7]	FF[4:7]	FF[0:3]

Functions that load ShC

Shift controls come from ShC except when BSEL.0 is 1 in the microinstruction that shifts

Shift controls come from FF when BSEL.0 is 1, and the source for B is changed to Q.

P = A[8:11] = number of bits to the left of the field
S = A[12:15] = number of bits in the field - 1

The values for RMask, LMask, and Shift Count are and'ed by 17-octal.
The 32-bit quantity SHA[0:15]..SHB[0:15] are left-cycled by the shift count and the right-most 16 bits are the shift data.

RF_ and WF_ are intended for use with "reasonable" values of P and S.

Derivation of Shift Controls

Figure 4

Shifter

D1Shifter.sil

9/5/79

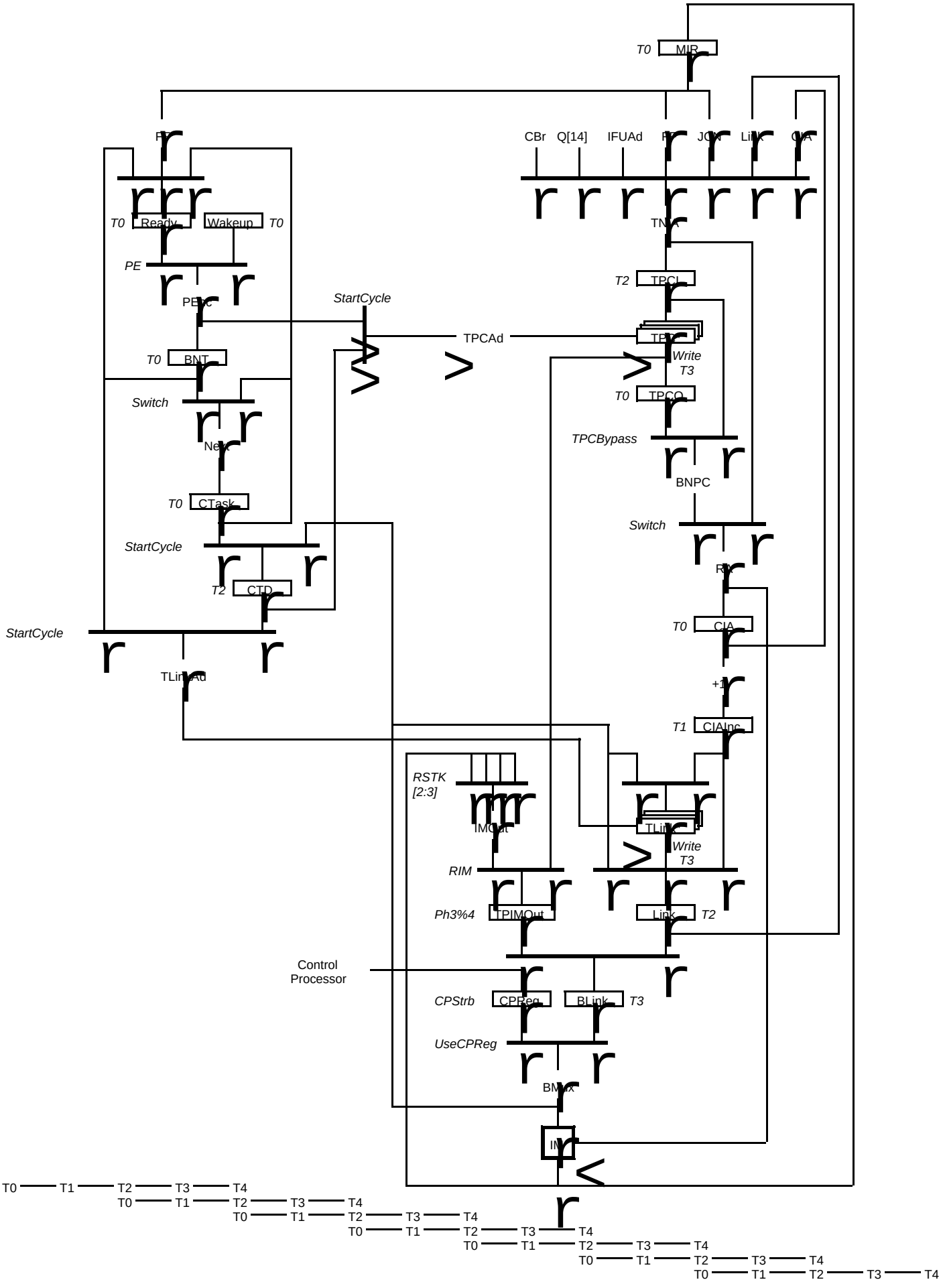


Figure 5
Control Section

D1Control.sil
9/5/79

JCN

0	1	2	3	4	5	6	7		TNIA:
1	0	ADDRESS BITS						Local Jump/Call	CIA[2:9] | JCN[2:7]
1	1	ADDRESS BITS						Global Call	CIA[2:3] | JCN[2:7] | 0 0 0 0 0 0
0	0	0	0	*ADDRESS BITS*				Long Jump/Call	CIA[2:3] | FF[0:7] | JCN[4:7]
0	*ADDRESS BITS # 000x*			*BRANCH CONDITION # 111*				Conditional Jump/Call	CIA[2:9] | JCN[1:2] | 0 | JCN[3:4] | R
0	1	*RETURN FUNCTION*		1	1	1		Return	Link[2:15]
0	0	1	*NEXT NUMBER*	1	1	1		IFU Jump	CIA[2:3] | InstrAddr[4:13] | JCN[3:4]
0	0	0	1	x	1	1	1	undefined	

2 3 4 5 6 7 8 9 10 11 12 13 14 15

R is result

A long, local, or conditional branch is a call iff, before any modification of TNIA by branch conditions or dispatches, TNIA[12:15] is 0; otherwise, it is a jump.

Conditional Branch

JCN[5:7]	*-or-*	*FF*	*BRANCH CONDITION*
0		60	ALU = 0
1		61	ALU < 0
2		62	Carry'
3		63	Cnt=0&-1 (decrement Cnt after testing)
4		64	R < 0
5		65	R odd
6		66	IOAtten' (non-emulator) -or- Reschedule (emulator)
--		67	Overflow

Return

JCN[2:4]	*FUNCTION*
0	Subroutine Return
1	unused
2	unused
3	unused
4	Read TPC
5	WriteTPC
6	Read Instruction Memory
	Address is in Link.
	Data appears on B[7:15] when B_Link executed in following microinstruction.
7	Write Instruction Memory
	Address is in Link. RSTK.3 is 1 to write the left half of IM, 0 to write the right half.

Loaded into Link by Call, Return, or IFUJump

CIA[2:9] | CIA[10:15] + 1

2 3 4 5 6 7 8 9 10 11 12 13 14 15

RSTK[2:3]

0	RSTK.0	RSTK.1	RSTK.2	RSTK.3	ALUF.0	ALUF.1	ALUF.2	ALUF.3	BSEL.0
1	Par.16	BSEL.1	BSEL.2	LC.0	LC.1	LC.2	ASEL.0	ASEL.1	ASEL.2
2	BLOCK	FF.0	FF.1	FF.2	FF.3	FF.4	FF.5	FF.6	FF.7
3	Par.17	JCN.0	JCN.1	JCN.2	JCN.3	JCN.4	JCN.5	JCN.6	JCN.7

Good (odd) parity is written if RSTK.1 is 0, else bad (even) parity is written.

The most significant bit of data is RSTK.2 and the least significant 16 bits are B[0:15].

RSTK.0	RSTK.1	RSTK.2	RSTK.3	ALUF.0	ALUF.1	ALUF.2	ALUF.3	BSEL.0	BSEL.1	BSEL.2	LC.0	LC.1	LC.2	ASEL.0	ASEL.1	ASEL.2
RSTK.2	B.0	B.1	B.2	B.3	B.4	B.5	B.6	B.7	B.8	B.9	B.10	B.11	B.12	B.13	B.14	B.15
BLOCK	FF.0	FF.1	FF.2	FF.3	FF.4	FF.5	FF.6	FF.7	JCN.0	JCN.1	JCN.2	JCN.3	JCN.4	JCN.5	JCN.6	JCN.7

Figure 6

Next Address Formation

D1Branching.sil
6/26/80

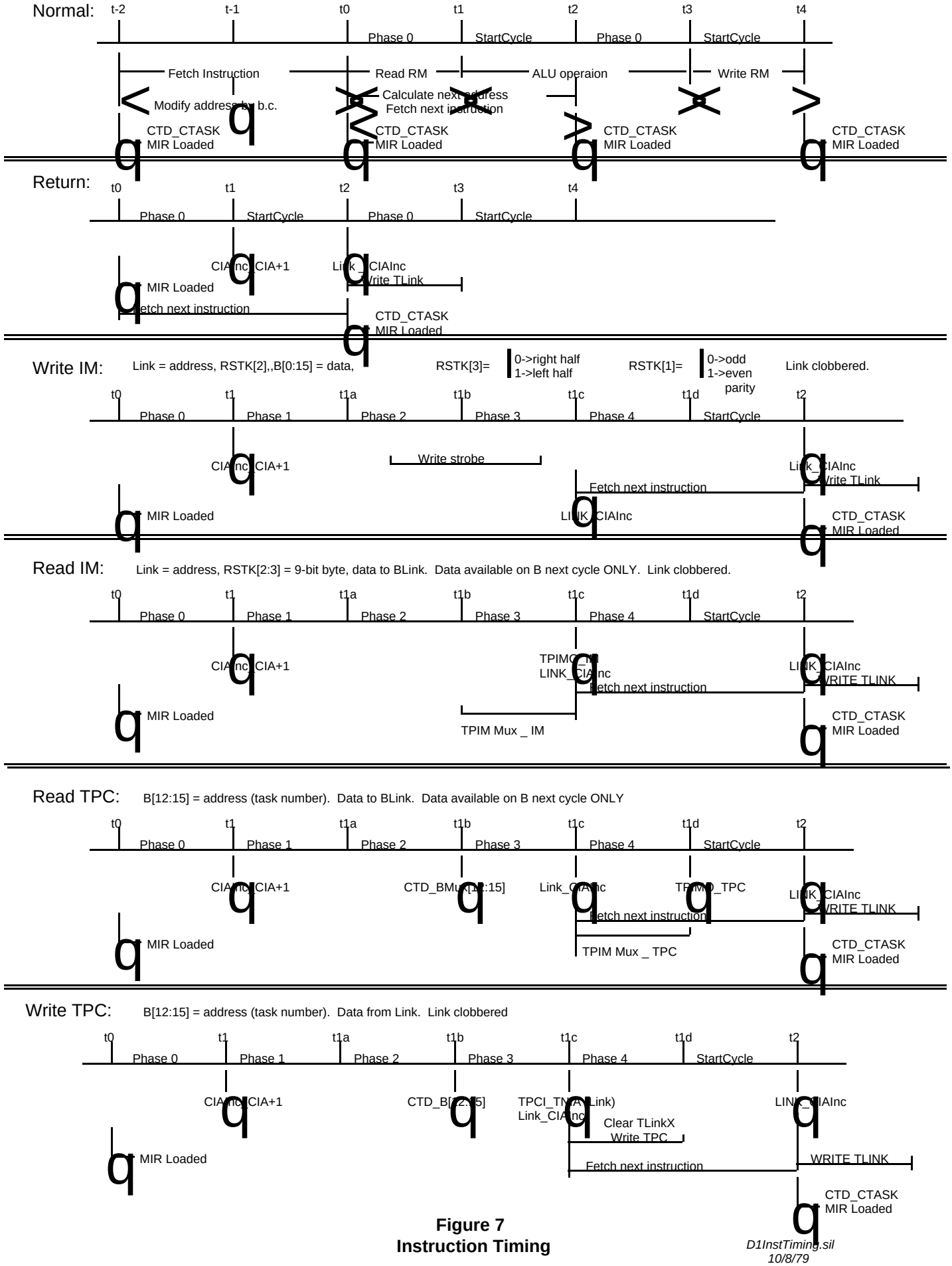


Figure 7
Instruction Timing

D1InstTiming.sil
10/8/79

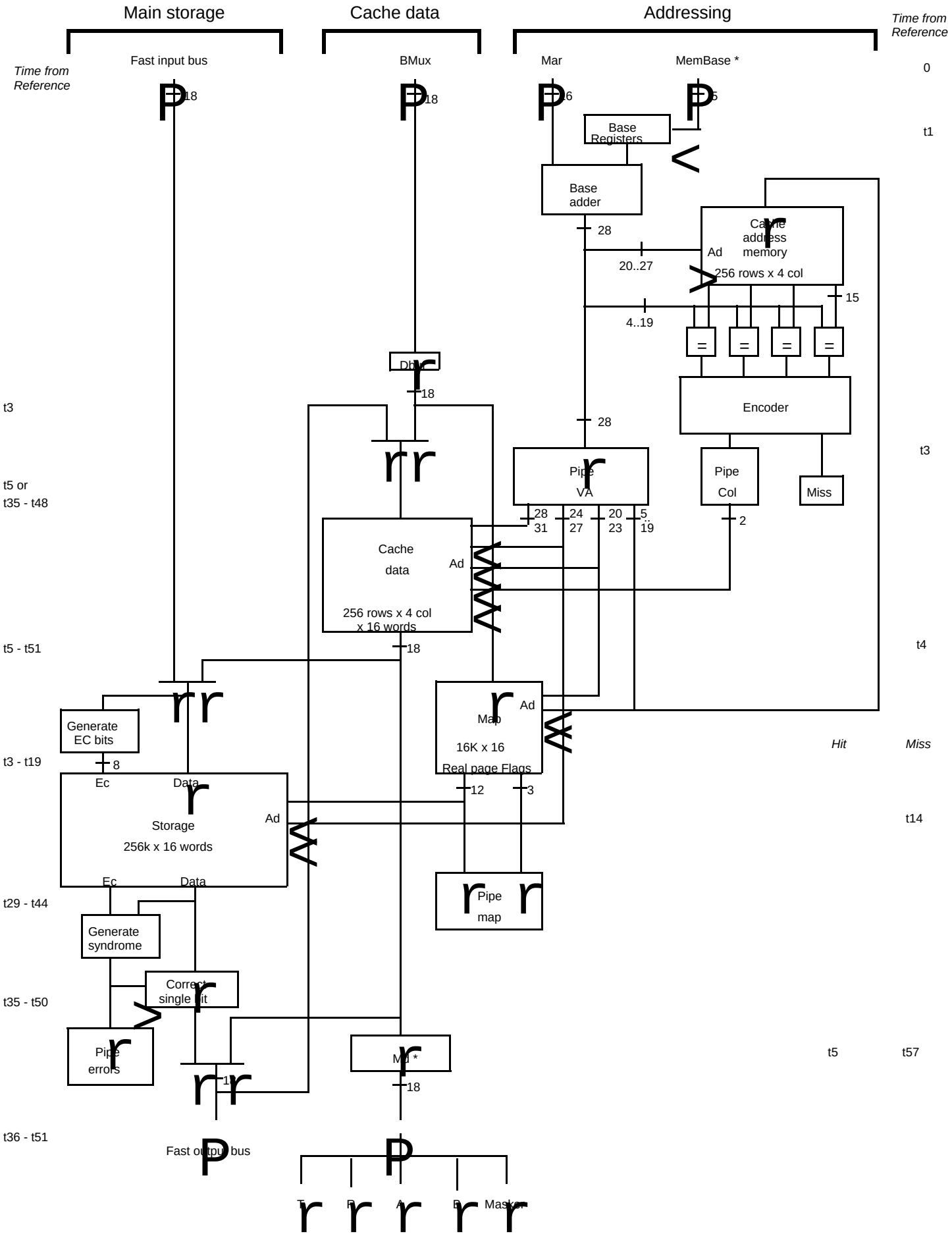


Figure 8
Overall Structure of the Memory System

D1Memory.sil
9/5/79

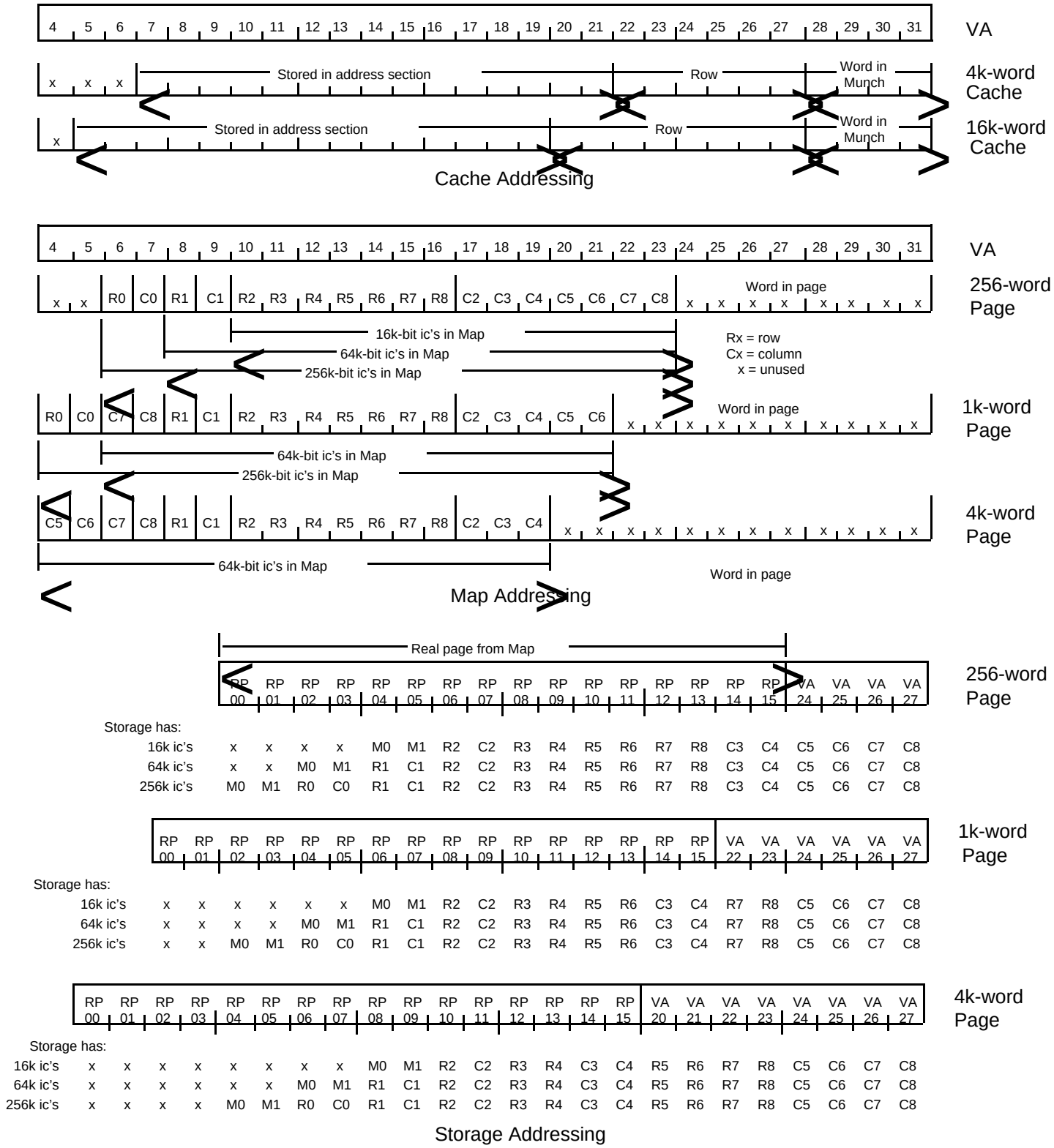


Figure 9

Cache, Map, and Storage Addressing

D1MemAddr.sil
8/31/79

a

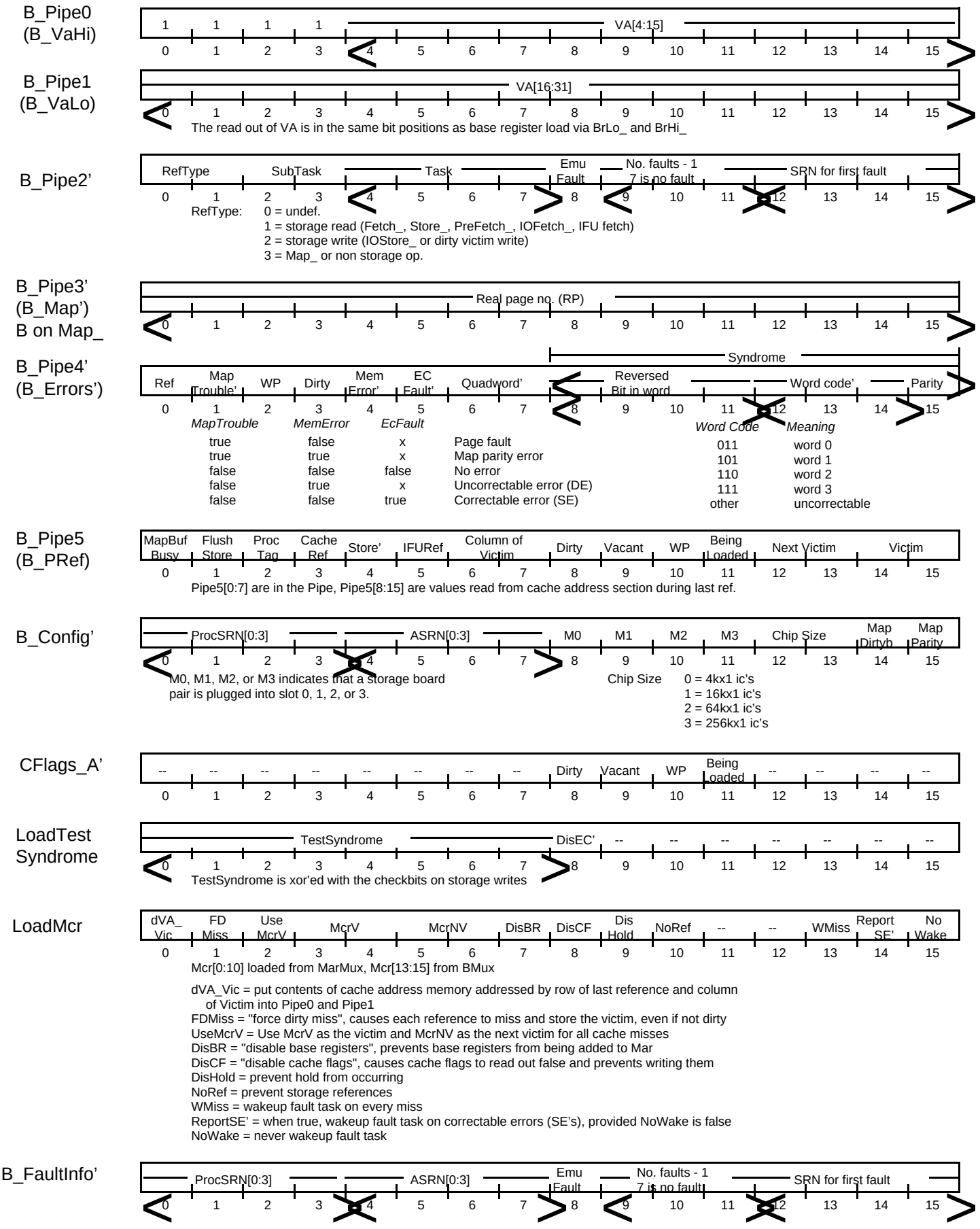
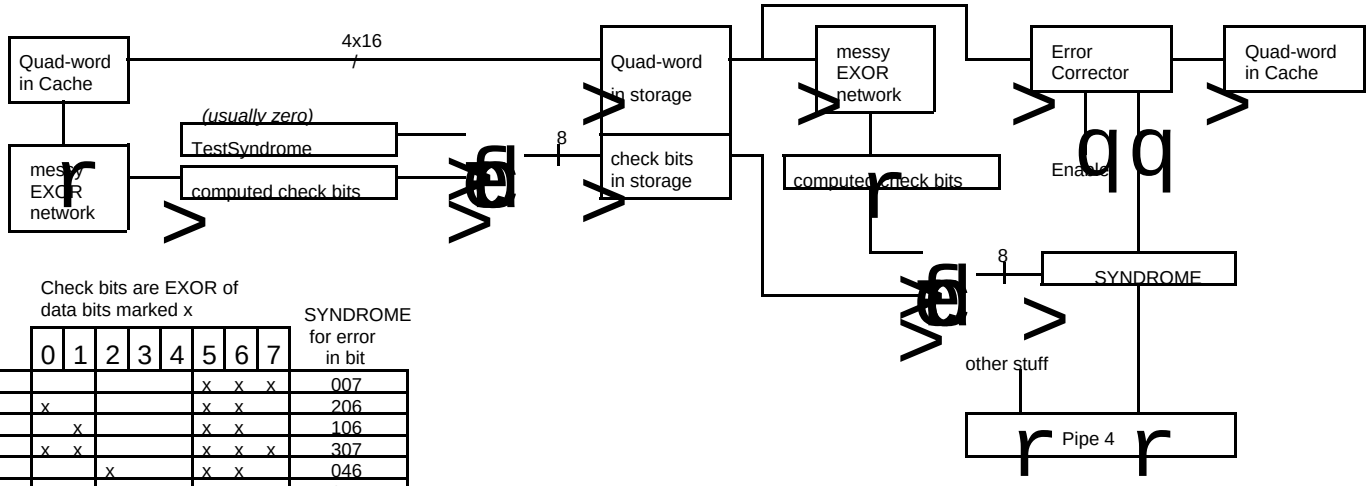


Figure 10
The Pipe and Other Memory Registers

D1MemReg.sil
6/26/80



Check bits are EXOR of data bits marked x

		0	1	2	3	4	5	6	7	SYNDROME for error in bit
WORD 0	00						x	x	x	007
	01	x					x	x		206
	02		x				x	x		106
	03	x	x				x	x	x	307
	04			x			x	x		046
	05	x		x			x	x	x	247
	06		x	x			x	x	x	147
	07	x	x	x			x	x		346
	08				x		x	x		026
	09	x			x		x	x	x	227
	10		x		x		x	x	x	127
	11	x	x		x		x	x		326
	12			x	x		x	x	x	067
	13	x		x	x		x	x		266
	14		x	x	x		x	x		166
	15	x	x	x	x		x	x	x	367
WORD 1	00					x		x	x	013
	01	x				x		x		212
	02		x			x		x		112
	03	x	x			x		x	x	313
	04			x		x		x		052
	05	x		x		x		x	x	253
	06		x	x		x		x	x	153
	07	x	x	x		x		x		352
	08				x	x		x		032
	09	x			x	x		x	x	233
	10		x		x	x		x	x	133
	11	x	x		x	x		x		332
	12			x	x	x		x	x	073
	13	x		x	x	x		x		272
	14		x	x	x	x		x		172
	15	x	x	x	x	x		x	x	373
WORD 2	00					x	x		x	015
	01	x				x	x			214
	02		x			x	x			114
	03	x	x			x	x		x	315
	04			x		x	x			054
	05	x		x		x	x		x	255
	06		x	x		x	x		x	155
	07	x	x	x		x				354
	08				x	x	x			034
	09	x			x	x	x		x	235
	10		x		x	x	x		x	135
	11	x	x		x	x	x			334
	12			x	x	x	x		x	075
	13	x		x	x	x	x			474
	14		x	x	x	x	x			174
	15	x	x	x	x	x	x		x	375
WORD 3	00					x	x	x		016
	01	x				x	x	x	x	217
	02		x			x	x	x	x	117
	03	x	x			x	x	x		316
	04			x		x	x	x	x	057
	05	x		x		x	x	x		256
	06		x	x		x	x	x		156
	07	x	x	x		x	x	x	x	357
	08				x	x	x	x	x	037
	09	x			x	x	x	x		136
	10		x		x	x	x	x		236
	11	x	x		x	x	x	x	x	337
	12			x	x	x	x	x		076
	13	x		x	x	x	x	x	x	277
	14		x	x	x	x	x	x	x	177
	15	x	x	x	x	x	x	x		376

Interpretation of SYNDROME

If SYNDROME is	and number of ones in Syndrome is	and furthermore	THEN
0	ALWAYS	ALWAYS	NO ERROR
Not 0	ODD	syndrome bits 4,5,6 have 2 or 3 ones on	SINGLE ERROR (data bit) Bits 4,5,6 give bad word: 4 5 6 | word 0 1 1 | 0 1 0 1 | 1 1 1 0 | 2 1 1 1 | 3 Bits 3,2,1,0 give bad bit: 3 2 1 0 | bit 0 0 0 0 | 00 0 0 0 1 | 01 ⋮ | ⋮ 1 1 1 0 | 14 1 1 1 1 | 15 Bad bit will be corrected if error correction is enabled
Not 0	ODD	exactly 1 one in Syndrome	SINGLE ERROR (check bit) Syndrome | bad check bit 200 | 0 100 | 1 040 | 2 020 | 3 010 | 4 004 | 5 002 | 6 001 | 7 No data bits will be changed.
Not 0	ODD	syndrome bits 4,5,6 have 0 or 1 ones on	TRIPLE ERROR!! (but no data bits will be changed) Syndrome is nonsense.
Not 0	EVEN	ALWAYS	DOUBLE ERROR No data bits will be changed. Syndrome is nonsense.

Figure 11

Error Correction

D1ErrCorr.sil
9/5/79

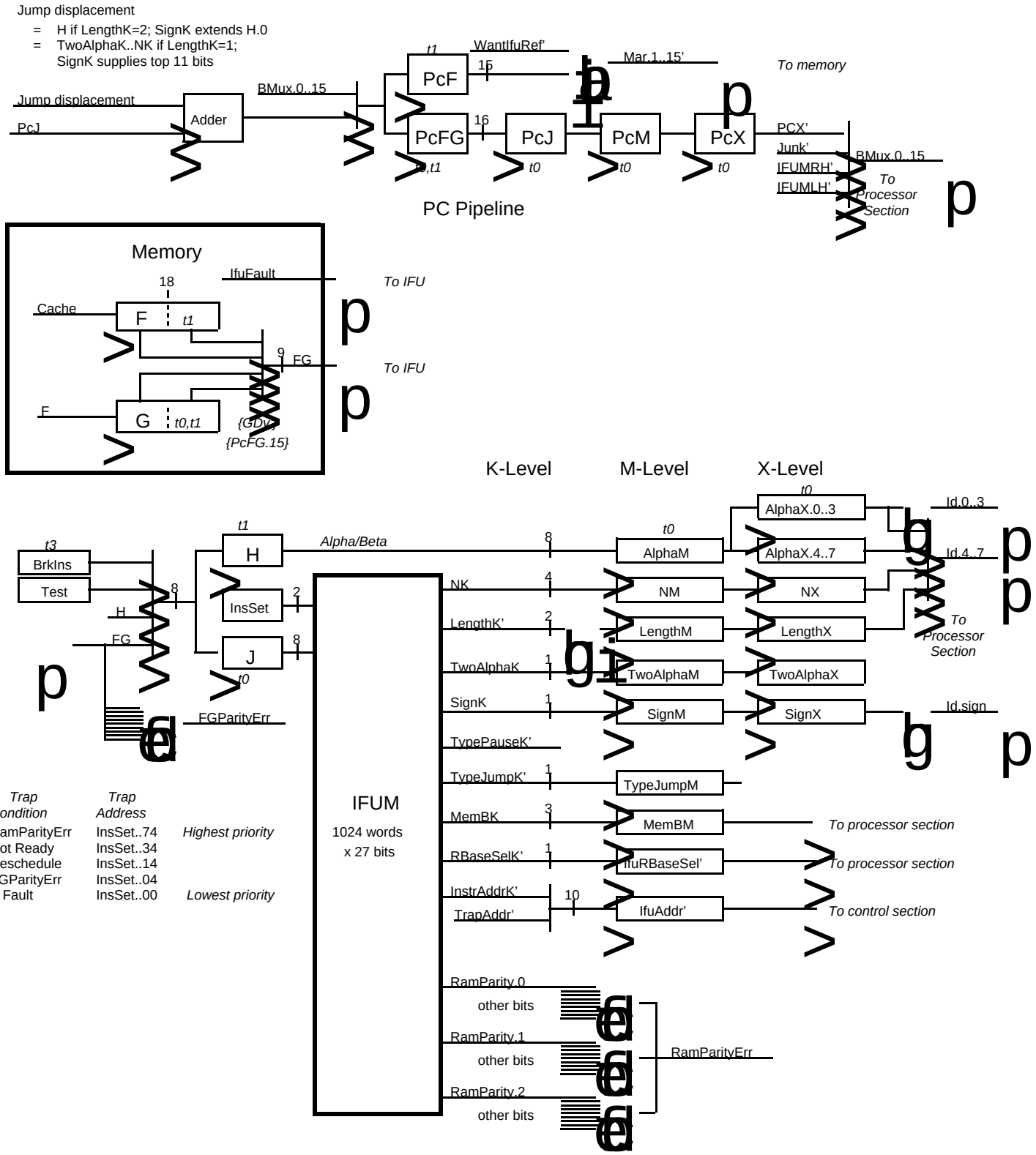


Figure 12

Instruction Fetch Unit Organization

D1IFU.sil

9/5/79

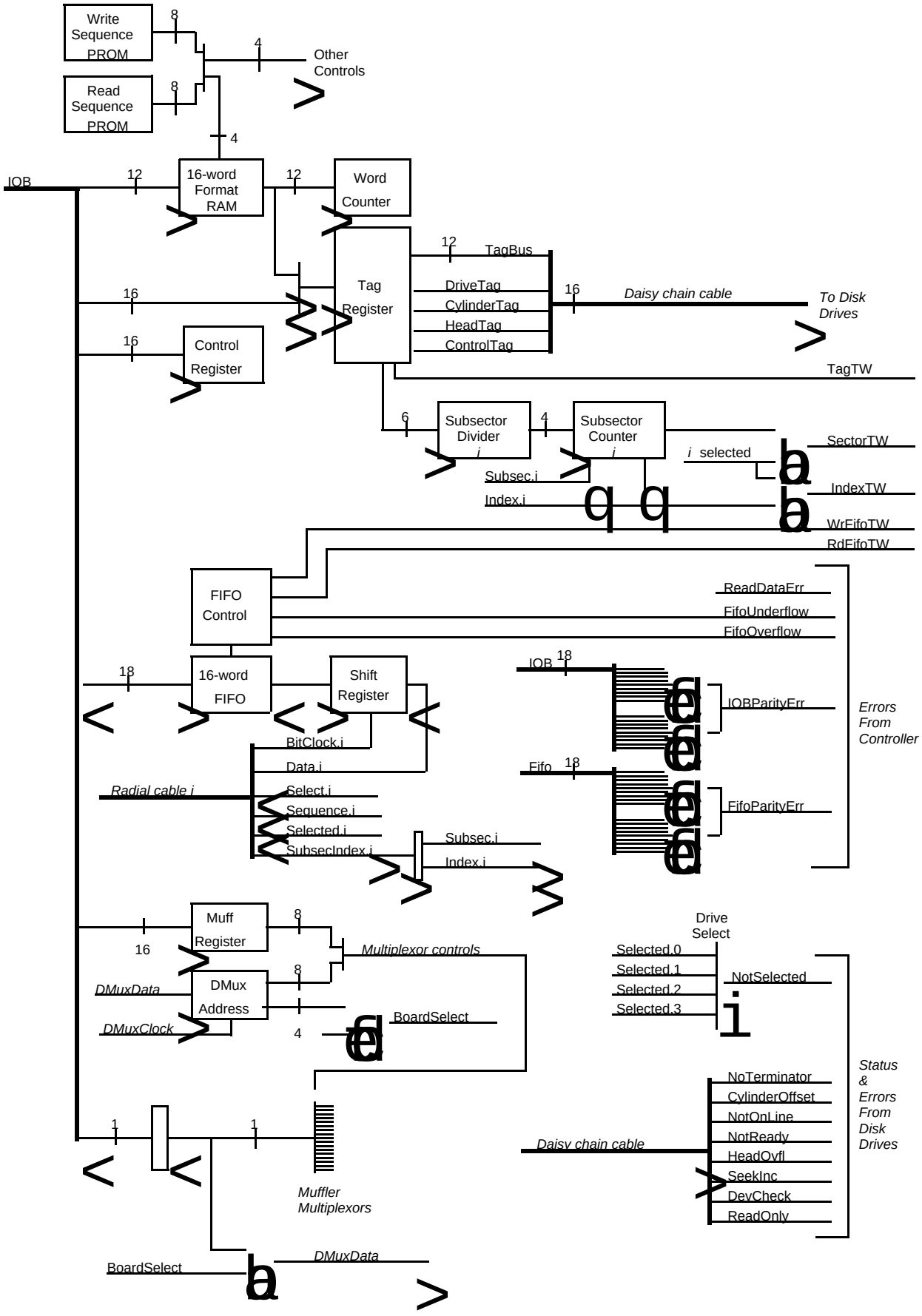
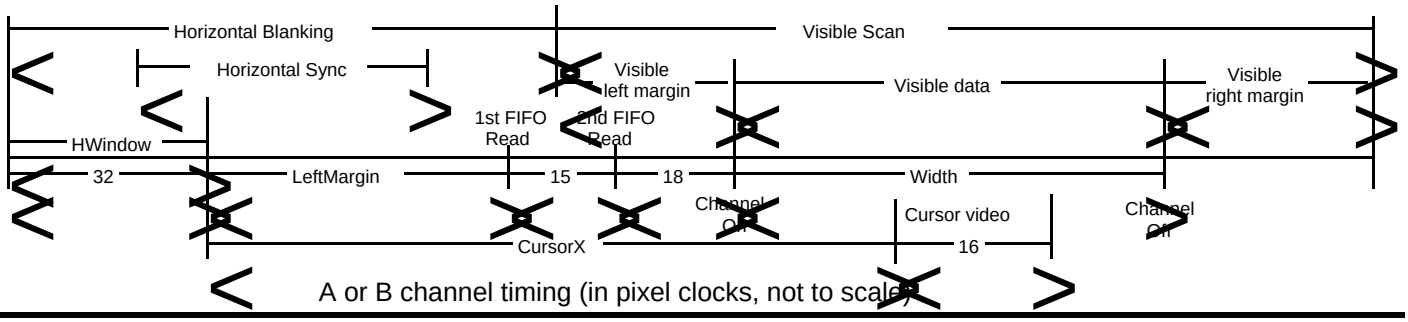
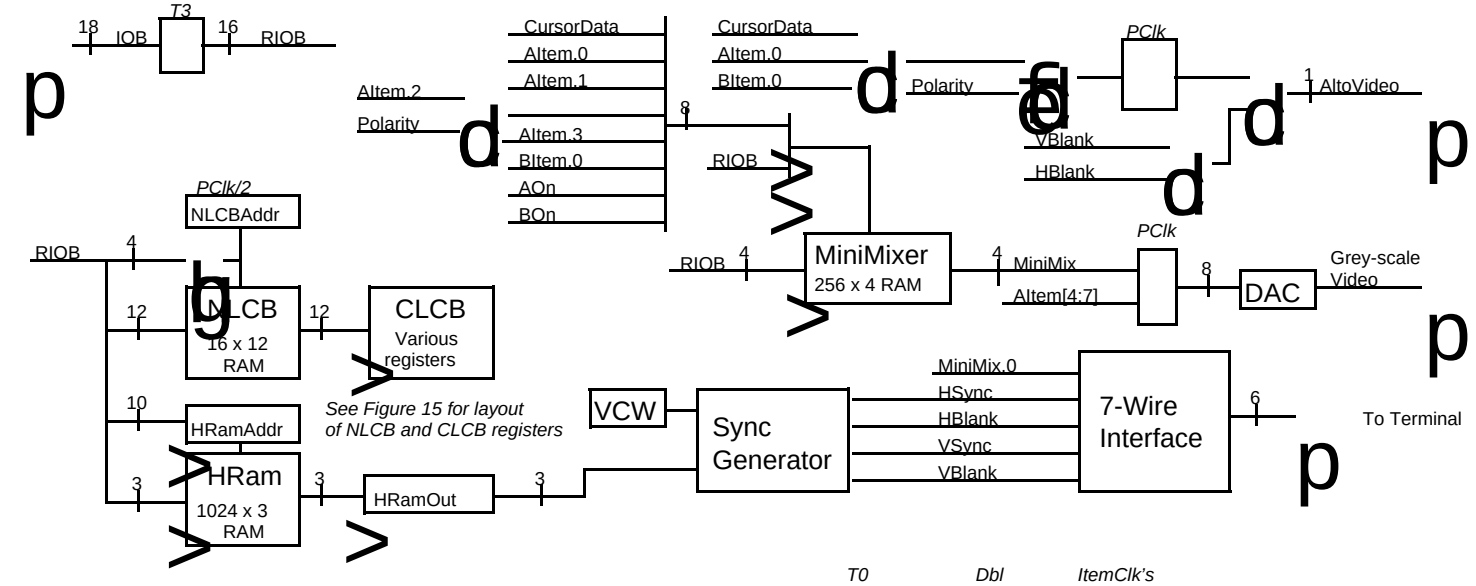


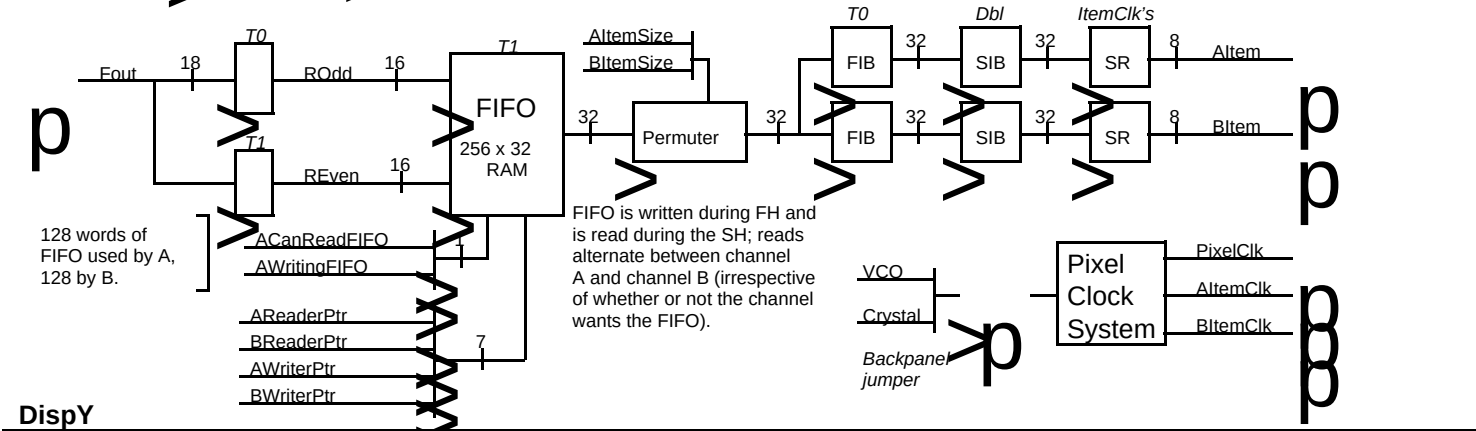
Figure 13
Disk Controller

D1Disk.sil
9/5/79



A or B channel timing (in pixel clocks, not to scale)





DispY

DispM (not including independent terminal interface)

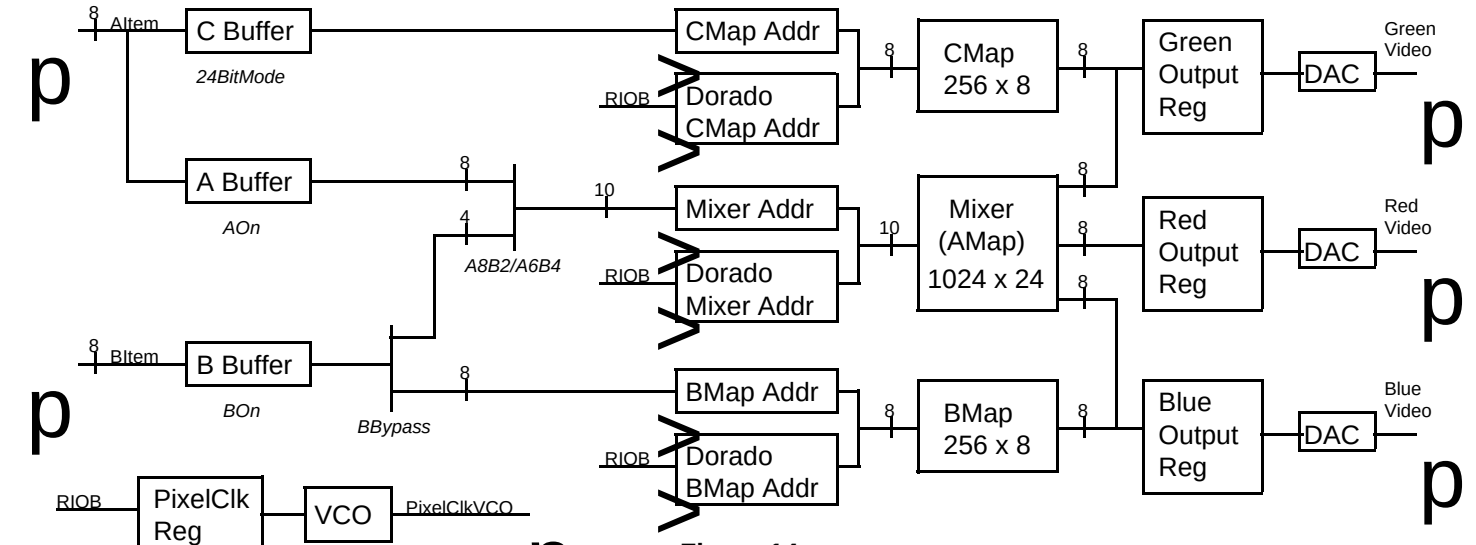
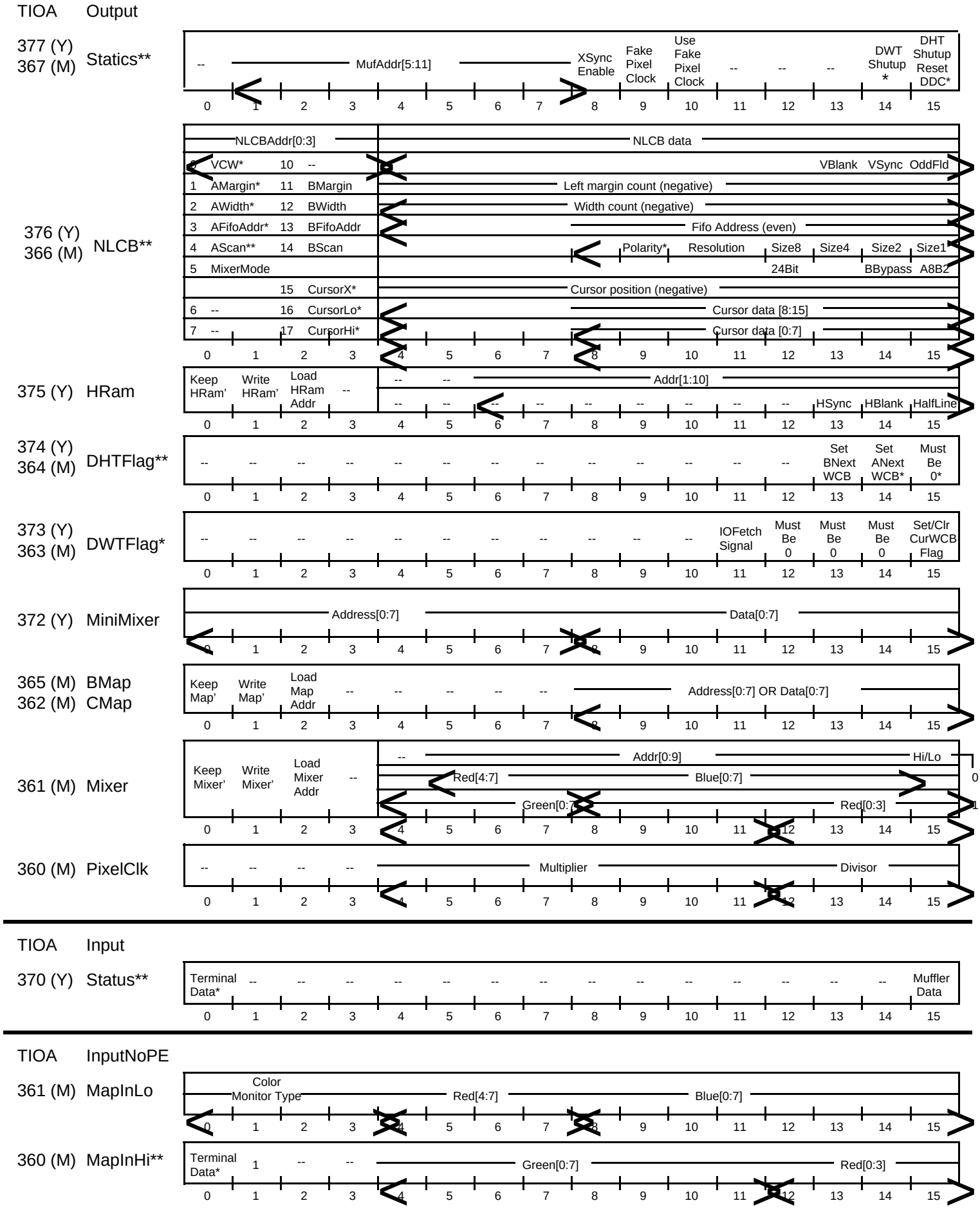


Figure 14
Display Controller

D1Display.sil
8/30/81

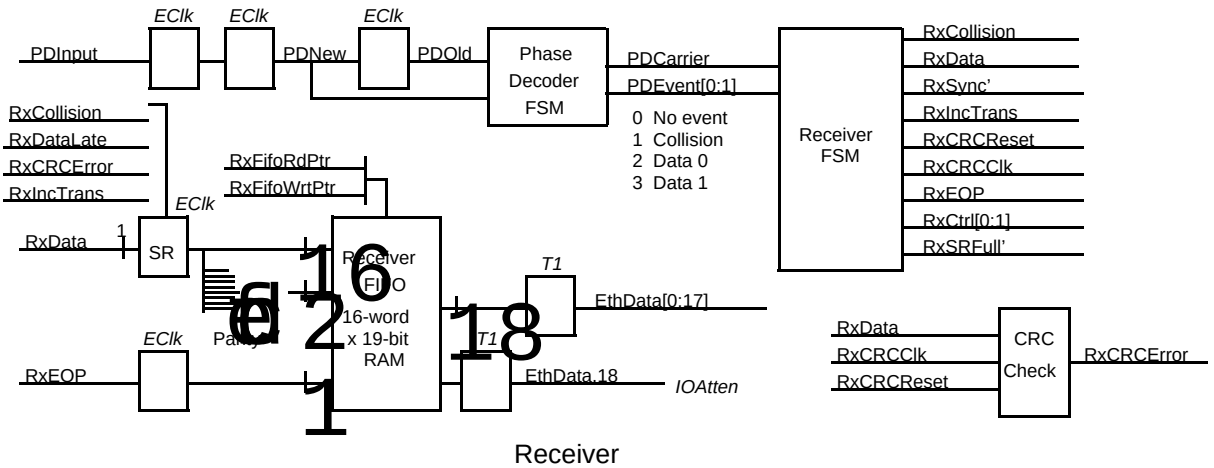


* Parallel registers DispY/DispM

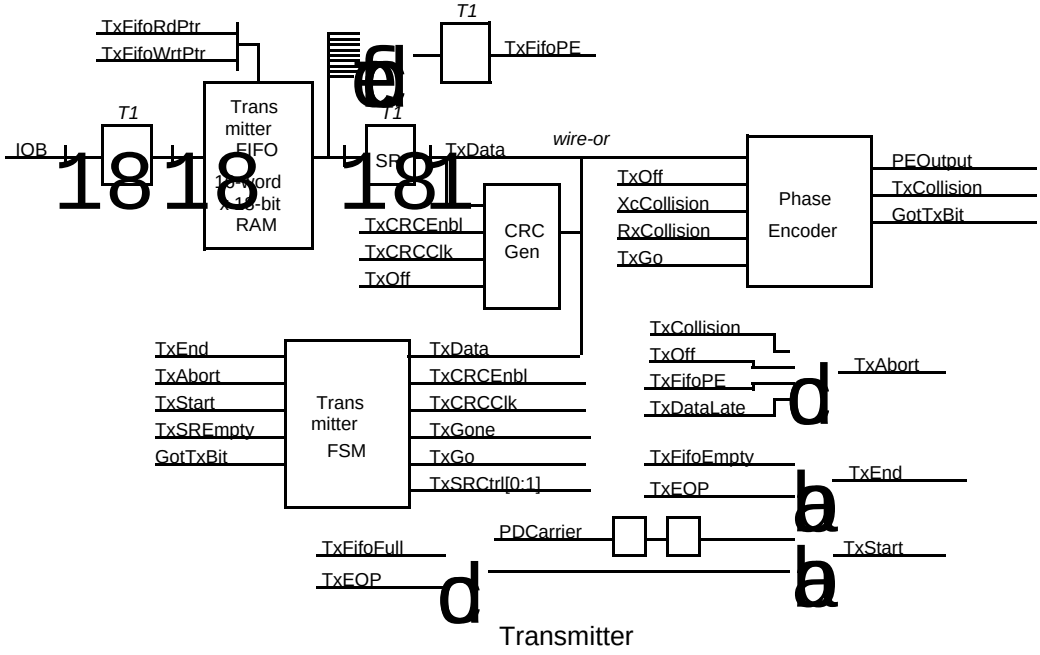
** Only starred bits or fields are used on DispM; all others are ignored

Figure 15
Display Controller IO Registers

D1DispReg.sil
8/29/81



Receiver



Transmitter

TIOA = 016
EthC
Output_B

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Tx Cmd Enbl'	TxOn	TxEOP	Tx Cnt Dwn	Rx Cmd Enbl'	RxOn	RxBOP'	--	Test Cmd Enbl'	Loop Back	Single Step	No Wake ups	Test Clock	Test Coll'	Test Data	Report Colls

TestCmdEnbl' enables setting of LoopBack, SingleStep, NoWakeups, TestClock, TestColl', TestData, and ReportColls
RxCmdEnbl' enables setting of RxOn and RxBOP'
TxCmdEnbl' enables setting of TxOn and TxEOP

EthC
Pd_Input

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
< Host Address >								RxOn	TxOn	Loop Back	TxColl	No Wake ups	Tx Data Late	Single Step	Tx Fifo PE

Host Address is set by backpanel jumpers

TIOA = 015
EthD
Pd_Input

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
--	--	--	--	--	--	--	--	Rx Collision	--	Rx Data Late	--	Rx CRC Error	--	--	Rx Inc Trans

(Receiver status word following end-of-packet)

Figure 16
Ethernet Controller

D1Ethernet.sil
8/30/81

BSEL Decodes

BSEL	Primary	External
0	Md	--
1	RM/STK	--
2	T	--
3	Q	Q_ B
4	0,,FF	
5	377,,FF	
6	FF,,0	
7	FF,,377	

LC Decodes

LC	Meaning
0	No action
1	T_ Pd
2	T_ Md, RM/STK_ Pd
3	T_ Md
4	RM/STK_ Md
5	T_ Pd, RM/STK_ Md
6	RM/STK_ Pd
7	T_ Pd, RM/STK_ Pd

FF Decodes

000-17	A[12:15]_ FF[4:7]
020	A_ RM/STK
021	A_ T
022	A_ Md
023	A_ Q
024	XorCarry
025	XorSavedCarry
026	Carry20
027	ModStkPBeforeW
030	--
031	ReadMap
032	Pd_ Input
033	Pd_ InputNoPE
034	RisId
035	TisId
036	Output_ B
037	FlipMemBase
040-57	Replace RSTK by FF[4:7] for write
060-67	Branch conditions
070	BigBDispatch_ B
071	BDispatch_ B
072	Multiply
073	Q_ B
074	--
075	TgetsMd
076	FreezeBC
077	Noop
100	PCF_ B
101	IFUTest_ B
102	IFUTick
103	RescheduleNow
104	--
105	MemBase_ B[3:7]
106	RBase_ B[12:15]
107	Pointers_ B
110-17	--
120-21	--
122	CFlags_ A'
123	BrLo_ A
124	BrHi_ A
125	LoadTestSyndrome
126	LoadMcr[A,B]
127	ProcSRN_ B[12:15]
130	InsSetorEvent_B
131	EventCntB_B
132	Reschedule
133	NoReschedule
134	IFUMRH_ B
135	IFUMLH_ B
136	IFUReset
137	BrkIns_ B
140	UseDMD
141	MidasStrobe_ B
142	TaskingOff

ASEL Decodes (FF is ok)

ASEL	FF[0:1]	Meaning
0	0	PreFetch_ RM/STK
	1	Map_RM/STK (emu/flt)
		IOFetch_RM (io)
	2	LongFetch_ RM/STK
	3	Store_ RM/STK
1	0	DummyRef_ RM/STK
	1	Flush_RM/STK (emu/flt)
		IOStore_RM (io)
	2	IFetch_ RM/STK
	3	Fetch_ RM/STK
2	0	Store_ Md
	1	Store_ Id
	2	Store_ Q
	3	Store_ T
3	0	Fetch_ Md
	1	Fetch_ Id
	2	Fetch_ Q
	3	Fetch_ T
4	--	A_ RM/STK
5	--	A_ Id
6	--	A_ T
7	--	Shift operation

ASEL Decodes (FF not ok)

ASEL	Meaning
0	Store_ RM/STK
1	Fetch_ RM/STK
2	Store_ T
3	Fetch_ T
4	A_ RM/STK
5	A_ Id
6	A_ T
7	Shift operation

RSTK Decodes for STK Operations

RSTK[0]: 0 = No ovfl/undfl check; 1 = Ovfl/undfl check

RSTK[1:3]	Meaning
0	No StkP change
1	StkP_ StkP+1
2	StkP_ StkP+2
3	StkP_ StkP+3
4	StkP_ StkP - 4
5	StkP_ StkP - 3
6	StkP_ StkP - 2
7	StkP_ StkP - 1

ALUFM Control Values

Logical Value	Addr	Meaning
1	16	NOT A
3		NOT A OR NOT B
5		NOT A OR B
7		A1 (all ones)
11		NOT A AND NOT B
13	14	NOT B
15		A XNOR B, A EQV B, A=B
17		A OR NOT B
21		NOT A AND B
23	10	A XOR B, A#B
25	0	B
27	7	A OR B
31	11	A0 (all zeroes)
33	15	A AND NOT B
35	6	A AND B
37		A

Arithmetic (no carry) Value	Addr	Meaning
0	1	A
6		2*A
14	2	A+B
22	5	A - B - 1
36	13	A - 1

Arithmetic (with carry) Value	Addr	Meaning
200	12	A+1
206		2*A+1
214	3	A+B+1
222	4	A - B
236		A

ALUFM addresses for operations in standard system microcode

ALUF Shift Decodes

ALUF[0:2]	Meaning
0	ShiftNoMask
1	ShiftLMask
2	ShiftRMask
3	ShiftBothMasks
4	ShMdNoMask
5	ShMdLMask
6	ShMdRMask
7	ShMdBothMasks

Derivation of Shift Controls

Field:	SHA	SHB	Count	RMask	LMask	
ShC bits:	2	3	4:7	8:11	12:15	
RF_ A	A[2]	A[3]	P+S+1	undefined	15-S	Shift controls come from Shc when BSEL[0]=0 in the microinstruction that shifts
WF_ A	A[2]	A[3]	16-P-S-1	16-P-S-1	P	
ShC_ B	B[2]	B[3]	B[4:7]	B[8:11]	B[12:15]	
	BSEL.1	BSEL.2	FF[4:7]	FF[4:7]	FF[0:3]	Shift controls come from FF when BSEL[0]=1, and the source for B is changed to Q

P=A[8:11]=number of bits to the left of the field
S=A[12:15]=number of bits in the field - 1

143	TaskingOn
144	StkP_ B[8:15]
145	RestoreStkP
146	Cnt_ B
147	Link_ B
150	Q lsh 1
151	Q rsh 1
152	TIOA[0:7]_ B[0:7]
153	--
154	Hold&TaskSim_ B
155	WF_ A
156	RF_ A
157	ShC_ A
160	B_ FaultInfo'
161	B_ Pipe0 (VaHi)
162	B_ Pipe1 (VaLo)
163	B_ Pipe2'
164	B_ Pipe3' (Map')

165	B_ Pipe4' (Errors')
166	B_ Config'
167	B_ Pipe5
170	B_ PCX'
171	B_EventCntA'
172	B_ IFUMRH'
173	B_ IFUMLH'
174	B_EventCntB'
175	B_ DBuf
176	B_ RWCPReg
177	B_ Link
200-17	RBase_ FF[4:7]
220-37	Replace RBase by FF[4:7] for write
240-47	TIOA[5:7]_ FF[5:7]
250-53	MemBaseX_ FF[6:7]
254-57	MemBX_ FF[6:7]

260-61	--
262	Pd_ ALUFMRW
263	Pd_ ALUFMEM
264	Pd_ Cnt
265	Pd_ Pointers
266	Pd_ TIOA&StkP
267	Pd_ ShC
270	Pd_ ALU rsh 1
271	Pd_ ALU rcy 1
272	Pd_ ALU brsh 1
273	Pd_ ALU arsh 1
274	Pd_ ALU lsh 1
275	Pd_ ALU lcy 1
276	Divide
277	CDivide
300-37	MemBase_ FF[3:7]
340-57	Cnt_ FF[4:7]
360-67	Wakeup[FF[4:7]]

Figure 17
Programmers' Crib Sheet

D1CribSheet.sil
9/26/79