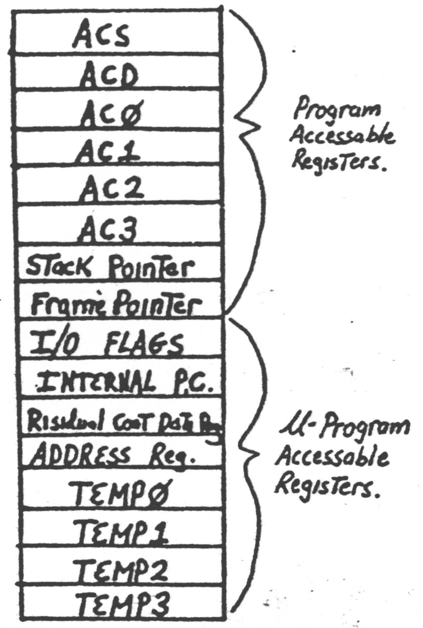
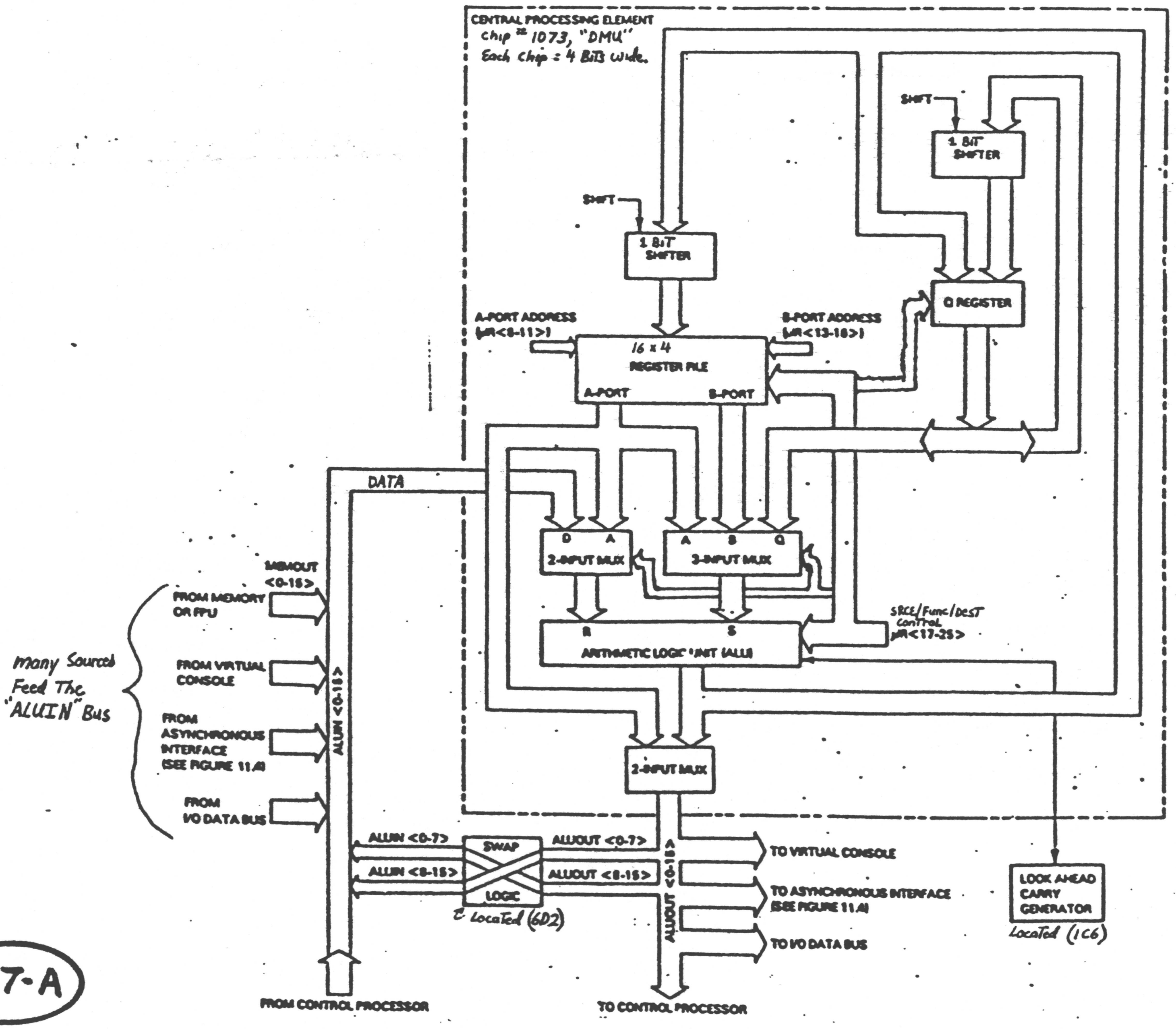
Data Minipulation Unit
Located SHEET(1D7)





MEMORY ADDRESSING (For Read or Write)

Current Micro-Cycle's ALUOUT Address Data is latched into "Memin Latches" (2D8) during time #4 of BMEMCLK. see CHART(2D1).

During Time #3, Next Micro-Cycle's CMEMO enables CMEMSTART (12B8) to (2A/B6). CMEMO also enables AND-NOR GATE 28-0 (2B5) into F/F 26-E.

At start of Time #4, BMEMCLK latches in ALUOUT Address into Latches and generates CPUMEMSTART (2A/B5) to (2B7) FOR start Pulse to Memory, and enables F/F 26-E (2B4) for "MEMDRIVE" to (2D7). Thus enabling the ADDRESS OUT to memory.

MEMORY WRITE

At the start of the next Micro-Cycle CPUMEMSTART is still high (2A/B5) during Time #5, and is applied to Driver 28-E (2B6/7) as well as to NAND 28-J (2A5). MEMDRIVE (2D7) is also still enabled at this time.

During Time #5, μIR1 (2B8) generates MEMWRITE (2B6) and enables NAND 20-J (2A5) to enable "Rightwrite" across inverter 20-Y (2A5) and out to AND-NOR GATE 20-O (2B5).

Data for writing to memory is alo now available on inputs on MEMIN Latches (2D8) from ALUOUT <6-15>

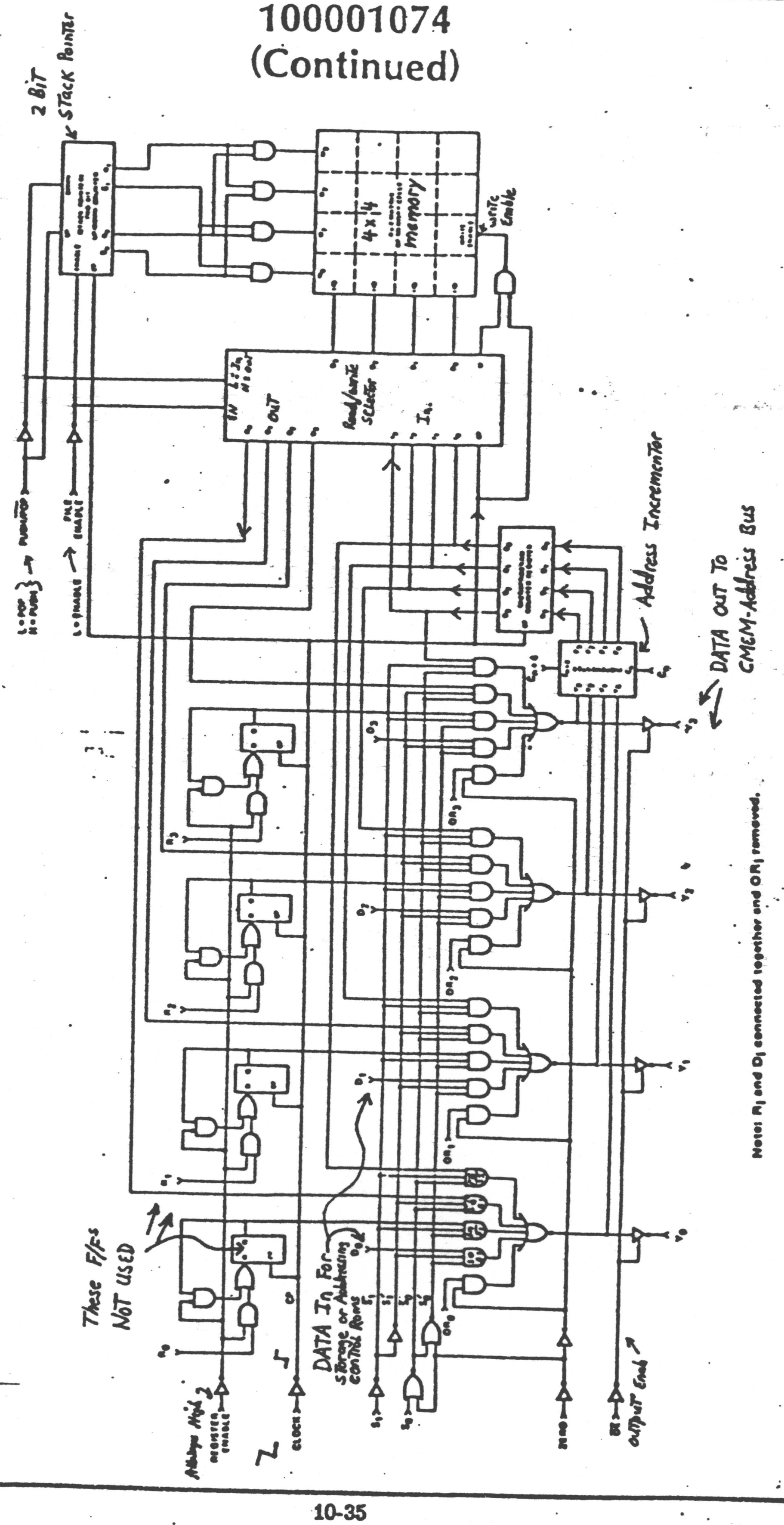
For ALUOUT <0-5> to be enabled onto latch inputs; Rightwrite (2A5) Low to NAND 28-J (11C5) thus disabling MAPOE High (11C/B4). MAPOE High, enables Driver 18-0 (11D5) thus enabling ALUOUT <0-5> to latches.

At Start of Time #6 CPUMEMSTART (2B7) goes Low. (Memory has already received the write CMD). ALUOUT DATA is latched (2D8) and F/F 26-E (2B4) stays set to enable the DATAOUT to memory via MEMDRIVE (2D/).

Micro-Sequencer

Located sheet (3D5)

100001074
(Continued)



MICROPROGRAM SEQUENCER BLOCK DIAGRAM

Note: R_i and D_i connected together and OR_i removed.

100001074
(Continued)

ADDRESS SELECTION

Pins 11 10

OCTAL	S_1	S_0	SOURCE FOR Y OUTPUTS	SYMBOL
0	L	L	Microprogram Counter	μPC
1	L	H	Register	REG
2	H	L	Push-Pop stack	STK0
3	H	H	Direct inputs	D_i

OUTPUT CONTROL

9 16

OR_i	$\overline{ZERO}$	$\overline{OE}$	Y_i
X	X	H	Z
X	L	L	L
H	H	L	H
L	H	L	Source selected by S_0 S_1

Z = High impedance

SYNCHRONOUS STACK CONTROL

19 20

$\overline{FE}$	PUP	PUSH-POP STACK CHANGE
H	X	No change
L	H	Increment stack pointer, then push current PC onto STK0
L	L	Pop stack (decrement stack pointer)

H = High
L = Low
X = Don't Care

OUTPUT AND INTERNAL NEXT-CYCLE REGISTER STATES

Pins: 11, 10, 19, 20 — Internal — 15-12

CYCLE	S_1, S_0, $\overline{FE}$, PUP	μPC	REG	STK0	STK1	STK2	STK3	Y_{OUT}	COMMENT	PRINCIPLE USE
N N+1	0 0 0 0 —	J J+1	K K	Ra Rb	Rb Rc	Rc Rd	Rd Ra	J —	Pop Stack	End Loop
N N+1	0 0 0 1 —	J J+1	K K	Ra J	Rb Ra	Rc Rb	Rd Rc	J —	Push μPC	Set-up Loop
N N+1	0 0 1 X —	J J+1	K K	Ra Ra	Rb Rb	Rc Rc	Rd Rd	J —	Continue	Continue
N N+1	0 1 0 0 —	J K+1	K K	Ra Rb	Rb Rc	Rc Rd	Rd Ra	K —	Pop Stack; Use AR for Address	End Loop
N N+1	0 1 0 1 —	J K+1	K K	Ra J	Rb Ra	Rc Rb	Rd Rc	K —	Push μPC; Jump to Address in AR	JSR AR
N N+1	0 1 1 X —	J K+1	K K	Ra Ra	Rb Rb	Rc Rc	Rd Rd	K —	Jump to Address in AR	JMP AR
N N+1	1 0 0 0 —	J Ra+1	K K	Ra Rb	Rb Rc	Rc Rd	Rd Ra	Ra —	Jump to Address in STK0; Pop Stack	RTS
N N+1	1 0 0 1 —	J Ra+1	K K	Ra J	Rb Ra	Rc Rb	Rd Rc	Ra —	Jump to Address in STK0; Push μPC	
N N+1	1 0 1 X —	J Ra+1	K K	Ra Ra	Rb Rb	Rc Rc	Rd Rd	Ra —	Jump to Address in STK0	Stack Ref (Loop)
N N+1	1 1 0 0 —	J D+1	K K	Ra Rb	Rb Rc	Rc Rd	Rd Ra	D —	Pop Stack; Jump to Address on D	End Loop
N N+1	1 1 0 1 —	J D+1	K K	Ra J	Rb Ra	Rc Rb	Rd Rc	D —	Jump to Address on D; Push μPC	JSR D
N N+1	1 1 1 X —	J D+1	K K	Ra Ra	Rb Rb	Rc Rc	Rd Rd	D —	Jump to Address on D	JMP D

X = Don't care, 0 = LOW, 1 = HIGH, Assume C_n = HIGH
Note: STK0 is the location addressed by the stack pointer.