

Stage/Label	Control Words	Comment
Fetch	PC_out, MAR_in, Read, Clear Y, Set Carry, F=ADD, Z_in	
	Z_out, PC_in, WMFC	
Decode	MDR_out, IR_in	
Branch to Instruction	μ -PC \leq PLA(IR)\$ [Source Fetching]::(Addressing Mode), [Destination Fetching]::(Addressing Mode), [No OP]::(Instruction), [Jump Subroutine]::(Instruction) [Branch]::Branch Offset, END [1]	This branches to the corresponding type of instruction to fetch the operands if there are any
Source Fetching		
Direct Register Mode	(Rsrc)_out, SRC_in	Source fetching is skipped in case of single operand instructions.
	μ -PC \leq PLA(IR)\$ [Destination Fetching]::(Addressing Mode) [2]	
Indirect Register Mode	(Rsrc)_out, MAR_in, Read, WMFC	
	μ -PC \leq PLA(IR)\$ [Source Fetching]::Move MDR to SRC	
Autoincrement Mode	(Rsrc)_out, MAR_in, Clear Y, Set Carry, F=ADD, Z_in	
	Z_out, (Rsrc)_in, WMFC	
	μ -PC \leq PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC	If indirect mode go to Indirect Mode, else go to Move MDR to SRC or Move MDR to DST
Autodecrement Mode	(Rsrc)_out, SET Carry, Clear Y, F=SUB, Z_in	
	Z_out, (Rsrc)_in, MAR_in, Read, WMFC	
	μ -PC \leq PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC	
Indexed	PC_out, MAR_in, Read, Clear Y, Set Carry, F=ADD, Z_in	
	Z_out, PC_in, WMFC	
	MDR_out, Y_in	
	(Rsrc)_out, F=ADD, Z_in	
	Z_out, MAR_in, Read, WMFC	
	μ -PC \leq PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC	
Indirect Mode	MDR_out, MAR_in, Read, WMFC	
Move MDR to SRC	MDR_out, SRC_in	
Destination Fetching		
Direct Register Mode	(Rdst)_out, DST_in	
	μ -PC \leq PLA(IR)\$ [Double Operand]::(Instruction) [3]	
Indirect Register Mode	(Rdst)_out, MAR_in, Read, WMFC	
	μ -PC \leq PLA(IR)\$ [Destination Fetching]::Move MDR to DST	
Autoincrement Mode	(Rdst)_out, MAR_in, Read, SET Carry, Clear Y, F=ADD, Z_in	
	Z_out, (Rdst)_in, WMFC	
	μ -PC \leq PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST	If indirect mode go to Indirect Mode, else go to Move MDR to SRC or Move MDR to DST
Autodecrement Mode	(Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in	
	Z_out, (Rdst)_in, MAR_in, Read, WMFC	
	μ -PC \leq PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST	
Indexed	PC_out, MAR_in, Read, Set Carry, Clear Y, F=ADD, Z_in	
	Z_out, PC_in, WMFC	
	MDR_out, Y_in	
	(Rdst)_out, F = ADD, Z_in	
	Z_out, MAR_in, Read, WMFC	
	μ -PC \leq PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST	
Indirect Mode	MDR_out, MAR_in, Read, WMFC	

Move MDR to DST	MDR_out, DST_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Single Operand]::(Instruction), [Double Operand]::(Instruction)	
Double Operand		
MOV SRC, DST	SRC_out, DST_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
ADD SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, CLR Carry, F=ADD	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
ADC SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, SET Carry, F=ADD	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
SUB SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, CLR Carry, F=SUB	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
SBC SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, SET Carry, F=SUB	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
AND SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, F=AND	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
OR SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, F=OR	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
XOR SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, F=XOR	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
CMP SRC, DST	SRC_out, Y_in	
	DST_out, Z_in, CLR Carry, F=SUB	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ END	
Single Operand		
INC DST	DST_out, SET Carry, Clear Y, F=ADD, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
DEC DST	DST_out, SET Carry, Clear Y, F=SUB, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
CLR DST	DST_out, Clear Y, F=AND, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
INV DST	DST_out, F=INV, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
LSR DST	DST_out, F=LSR, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
ROR DST	DST_out, F=ROR, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
ASR DST	DST_out, F=ASR, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
LSL DST	DST_out, F=LSL, Z_in	
	$\mu\text{-PC} \leftarrow \text{PLA}(\text{IR})\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	

ROL DST	DST_out, F=ROL, Z_in	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	
Branch		
Branch Offset	(IRoffset)_out, SRC_in	
	PC_out, DST_in	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ [Double Operand]::ADD SRC, DST	
No Operand		
HALT	It is implemented by a HALT flag raised in the PLA to set a signal to stop PC from updating	
RESET	Raise a reset signal in the PLA to reset all devices	
NOP	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ END	
Jump Subroutine		
JSR	PC_out, MAR_in, Read, Clear Y, Set Carry, F=ADD, Z_in	Here we rewrote the Fetch μ -program to save the clock cycle resulting from μ -branching, with each execution of the Fetch μ -program, trading off two words in the control store.
	Z_out, PC_in, WMFC	
	MDR_out, SRC_in	
	PC_out, MDRin	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ PUSH	
JSR after PUSH	(IRAddress)_out, PC_in	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ END	
INTERRUPT	STATUS_out, MDRin	ed we cannot determine which instance to branch back to based solely on the instruction. There
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ PUSH	
INTERRUPT after PUSH	PC_out, MDRin	
	SP_out, Clear Y, Set Carry, F=SUB, Z_in	This is another PUSH microprogram, but we do not know how to branch back from PUSH
	Z_out, SP_in, MAR_in, Write, WMFC	
	(InterruptAddress)_out, PC_in	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ END	
Start IRET/RTS/POP PC	SP_out, MAR_in, Read, Clear Y, Set Carry, F=ADD, Z_in	
	Z_out, SP_out, WMFC	
	MDR_out, PC_in	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ END, Continue IRET	
Continue IRET	SP_out, MAR_in, Read, Clear Y, Set Carry, F=ADD, Z_in	This is a POP microprogram but we do not know how to branch back here
	Z_out, SP_out, WMFC	
	MDR_out, STATUS_in	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ END	
PUSH	SP_out, Clear Y, Set Carry, F=SUB, Z_in	
	Z_out, SP_in, MAR_in, Write, WMFC	
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ INTERRUPT after PUSH, JSR after PUSH	
Move Z back to Rdst		
Direct Register Mode	Z_out, (Rdst)_in	etermined by a decoder, controlled by IR, which sets the enable line for the corresponding destin
	$\mu\text{-PC} \leq \text{PLA(IR)}\$$ END	
Indirect Write Mode	Z_out, MDRin, Write, WMFC	
End		
END	Send END signals	

[1] If double or single operand:
 branch to corresponding addressing mode
else if no op/branch/jsr
 branch to no op/branch/jsr

For HALT: raise HALT signal
For NOP: branch to Fetch again
For RTS: jump directly to POP

[2] Jumps to instruction microprogram using PLA

[3] Jumps to instruction microprogram using PLA