Fetch PC.Out, MAKI.In, Med. Clear Y, Set Carry, F-ADD, Z.In Z.Out, PC.In, MWPC Decode NUL.out, IR.In p=Cc = PLAI_EIS Source Fetching]: (Addressing Mode) [Bource Fetching]: (Addressing Mode) [Branch to Instruction I Jump Bubroutine]: (Instruction) I Branch: Seamen of Fetching]: (Addressing Mode) [Branch: Petching]: (Addressing Mode) [Branch: Seamen of Fetching]: (Addressing Mode) [Branch: Seamen of Fetching of Fetc	
Decode MPR. cut. TR. in cut. Companies Decode Dec	
Pinc of PLAIRINS Source Fetching : (Addressing Mode), Destination Fetching : (Addressing Mode), Destin	
Source Fetching ::(Addressing Mode), Destination Des	
Direct Register Mode (Rarc)_out, SRC_in Source Fetching]::(Addressing Mode) [2] 1 Source Fetching is skipped in case of single operand instruction p-PC = PLA(IRS) [Destination Fetching]::(Addressing Mode) [2] 1	here are any
PFC == PLA(IR)\$ [Destination Fetching]::(Addressing Mode) [2] 1	
PFC <= PLA(IR)S Destination Fetching ::(Addressing Mode) 2 1	
Indirect Register Mode	
##PC <= PLA(IR)\$ [Source Fetching]::Move MOR to SRC Autoincrement Mode (Rerc)_out, MAR_In, Clear Y, Set Carry, F=ADD, Z_in ##PC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC ##PC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC ##Indirect Mode Autodecrement Mode P-DC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode P-DC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode P-DC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode P-DC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode P-DC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode MDR_Out, SRC in Destination Fetching -PC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Register Mode Rdst]_out, DST_in Destination Fetching P-DC <= PLA(IR)\$ [Double Operand]::[Instruction) [3] Indirect Register Mode P-DC <= PLA(IR)\$ [Double Operand]::[Instruction) [3] Autoincrement Mode P-DC <= PLA(IR)\$ [Double Operand]::Indirect Mode, Move MDR to DST P-DC <= PLA(IR)\$ [Destination Fetching]::Move MDR to DST P-DC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST P-DC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST P-DC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST P-DC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST P-DC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST P-DC <	
Autoincrement Mode (Rsrc)_out, MAR_in, Clear Y, Set Carry, F=ADD, Z_in Z_out, (Rsrc)_in, WMFC µ PPC = PLA(IR)S [Source Fetching]::Indirect Mode, Move MDR to SRC 1 If indirect mode go to indirect Mode, else go to Move MDR to SRC or Move Autodecrement Mode (Rsrc)_out, SET Carry, Clear Y, F=SUB, Z_in Z_out, (Rsrc)_in, MAR_in, Read, WMFC µ PC = PLA(IR)S [Source Fetching]::Indirect Mode, Move MDR to SRC 1 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 Indirect Mode MDR_out, MAR_in, Read, WMFC Indirect Mode MDR_out, MAR_in, Read, WMFC Direct Register Mode (Rdst)_out, DST_in µ PC = PLA(IR)S [Source Fetching]::Indirect Mode Nove MDR to DST Autoincrement Mode (Rdst)_out, MAR_in, Read, WMFC µ PC = PLA(IR)S [Double Operand]::(Instruction) [3] 2 Indirect Register Mode (Rdst)_out, MAR_in, Read, WMFC µ PC < PLA(IR)S [Double Operand]::(Instruction) [3] Autoincrement Mode (Rdst)_out, MAR_in, Read, WMFC µ PC < PLA(IR)S [Double Operand]::Move MDR to DST Z_out, (Rdst)_in, WMFC Z_out, (Rdst)_in, WMFC Autoincrement Mode (Rdst)_out, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_out, FRC Carry, Clear Y, F=ADD, Z_in Autoincrement Mode Autodecrement Mode	
Z_OUT, (RSrC)_in, WMFC \[\begin{array}{cccccccccccccccccccccccccccccccccccc	
P-PC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC	
Autodecrement Mode (Rsrc)_out, SET Carry, Clear Y, F=SUB, Z_in	IDR to DST
Z_out, (Rsrc)_in, MAR_in, Read, WMFC	
Indexed	
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 <= PLA(IR)S [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode Move MDR to SRC MDR_out, MAR_in, Read, WMFC Move MDR to SRC Direct Register Mode (Rdst)_out, DST_in µ-PC <= PLA(IR)S [Double Operand]::(Instruction) [3] Indirect Register Mode (Rdst)_out, MaR_in, Read, WMFC µ-PC <= PLA(IR)S [Destination Fetching]::Move MDR to DST Autoincrement Mode (Rdst)_out, MaR_in, Read, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_out, MaR_in, Read, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_out, MaR_in, Read, SET Carry, Clear Y, F=SDB, Z_in Z_out, SET Carry, Clear Y, F=SUB, Z_in	
Z_out, PC_in, WMFC MR_out, Y_in (Rsrc_out, F=ADD, Z_in Z_out, MAR_in, Read, WMFC µ-PC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC Indirect Mode Mor_out, MAR_in, Read, WMFC Move MDR to SRC Mor_out, SRC_in Destination Fetching Direct Register Mode (Rdst)_out, DST_in µ-PC <= PLA(IR)\$ [Double Operand]::(Instruction) [3] Indirect Register Mode (Rdst)_out, MAR_in, Read, WMFC µ-PC <= PLA(IR)\$ [Double Operand]::Move MDR to DST Autoincrement Mode (Rdst)_out, MAR_in, Read, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_out, MAR_in, Read, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC	
MDR_out, Y_in (Rsrc)_out, F=ADD, Z_in Z_out, MAR_in, Read, WMFC µ=PC <= 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 <= PLA(IR)\$ [Double Operand]::(Instruction) [3] [Rdst)_out, MAR_in, Read, WMFC µ=PC <= 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 <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ=PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ=PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ=PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_in, WMFC µ=PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST Z_out, (Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in	
Z_out, MAR_in, Read, WMFC	
Z_out, MAR_in, Read, WMFC	
μ-PC <= PLA(IR)\$ [Source Fetching]::Indirect Mode, Move MDR to SRC	
Move MDR to SRC MDR_out, SRC_in Destination Fetching Direct Register Mode (Rdst)_out, DST_in	
Move MDR to SRC MDR_out, SRC_in Destination Fetching Direct Register Mode (Rdst)_out, DST_in	
Direct Register Mode (Rdst)_out, DST_in	
μ-PC <= PLA(IR)\$ [Double Operand]::(Instruction) [3] 2 Indirect Register Mode (Rdst)_out, MAR_in, Read, WMFC	
Indirect Register Mode (Rdst)_out, MAR_in, Read, WMFC \(\buildrel{\pu}-PC <= PLA(IR)\\$ [Destination Fetching]::Move MDR to DST 2 Autoincrement Mode (Rdst)_out, MAR_in, Read, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_in, WMFC \(\buildrel{\pu}-PC <= PLA(IR)\\$ [Destination Fetching]::Indirect Mode, Move MDR to DST 2 If indirect mode go to Indirect Mode, else go to Move MDR to SRC or Move Autodecrement Mode (Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in	
μ-PC <= 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 <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST 2 If indirect mode go to Indirect Mode, else go to Move MDR to SRC or Move Autodecrement Mode (Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in	
Autoincrement Mode (Rdst)_out, MAR_in, Read, SET Carry, Clear Y, F=ADD, Z_in Z_out, (Rdst)_in, WMFC µ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST (Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in (Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in	
Z_out, (Rdst)_in, WMFC	
μ-PC <= PLA(IR)\$ [Destination Fetching]::Indirect Mode, Move MDR to DST 2 If indirect mode go to Indirect Mode, else go to Move MDR to SRC or Mo	
Autodecrement Mode (Rdst)_out, SET Carry, Clear Y, F=SUB, Z_in	
(117)-117-117-117-117-117-117-117-117-117	IDR to DST
7 out (Rdst) in MAR in Read WMFC	
Z_VOC, (NOOC/_III, MAN_III, Neau, MM V	
μ-PC <= 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 <= 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	
μ-PC <= PLA(IR)\$ [Single Operand]::(Instruction), [Double Operand]::(Instruction)	
Double Operand	
MOV SRC, DST SRC_out, DST_in	

	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
ADD SRC, DST	SRC_out, Y_in		
	DST_out, Z_in, CLR Carry, F=ADD		
	μ -PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
ADC SRC, DST	SRC_out, Y_in		
	DST_out, Z_in, SET Carry, F=ADD		
	$\mu\text{-PC} \leftarrow PLA(IR)$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
SUB SRC, DST	SRC_out, Y_in		
	DST_out, Z_in, CLR Carry, F=SUB		
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
SBC SRC, DST	SRC_out, Y_in		
	DST_out, Z_in, SET Carry, F=SUB		
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
AND SRC, DST	SRC_out, Y_in		
	DST_out, Z_in, F=AND		
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
OR SRC, DST	SRC_out, Y_in		
·	DST_out, Z_in, F=OR		
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
XOR SRC, DST	SRC_out, Y_in		
•	DST_out, Z_in, F=XOR		
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	4	
CMP SRC, DST	SRC_out, Y_in		
	DST_out, Z_in, CLR Carry, F=SUB		
	μ-PC <= PLA(IR)\$ END	4	
	Single Operand		
INC DST	DST_out, SET Carry, Clear Y, F=ADD, Z_in		
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
DEC DST	DST_out, SET Carry, Clear Y, F=SUB, Z_in		
220 201	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
CLR DST	DST_out, Clear Y, F=AND, Z_in		
OLK DOT	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
INV DST	DST_out, F=INV, Z_in	3	
144 031	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
LOD DOT		3	
LSR DST	DST_out, F=LSR, Z_in	3	
DOD DOT	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
ROR DST	DST_out, F=ROR, Z_in		
AOD DOT	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
ASR DST	DST_out, F=ASR, Z_in	_	
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
LSL DST	DST_out, F=LSL, Z_in		
	μ -PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
ROL DST	DST_out, F=ROL, Z_in		
ROL DST	$\mu\text{-PC} \leftarrow PLA(IR)$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
ROL DST		3	
ROL DST Branch Offset	$\mu\text{-PC} \leftarrow PLA(IR)$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode	3	
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode Branch	3	
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode Branch (IRoffset)_out, SRC_in	1	
	μ-PC <= PLA(IR)\$ [Move Z to Rdst]::Direct Register Mode, Indirect Write Mode Branch (IRoffset)_out, SRC_in PC_out, DST_in		

RESET	Raise a reset signal in the PLA to reset all devices		
NOP	μ-PC <= PLA(IR)\$ END	1	
	Jump Subroutine		
JSR	PC_out, MAR_in, Read, Clear Y, Set Carry, F=ADD, Z_in		
	Z_out, PC_in, WMFC		Here we rewrote the Fetch μ-program to save the clock cycle resulting from μ-branching, with
	MDR_out, SRC_in		each execution of the Fetch μ -program, trading off two words in the control store.
	PC_out, MDRin		
	μ-PC <= PLA(IR)\$ PUSH	1	
JSR after PUSH	(IRAddress)_out, PC_in		
	μ-PC <= PLA(IR)\$ END	3	
INTERRUPT	STATUS_out, MDRin		led we cannot determine which instance to branch back to based solely on the instruction. There
	μ-PC <= PLA(IR)\$ PUSH	1	·
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		This is another POSH inicroprogram, but we do not know now to branch back from POSH
	(InterruptAddress)_out, PC_in		
	μ-PC <= PLA(IR)\$ END	3	
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		
	$\mu\text{-PC}$ <= PLA(IR)\$ END, Continue IRET	1	
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		This is a 1-O1 microprogram but we do not know now to branch back here
	MDR_out, STATUS_in		
	μ-PC <= PLA(IR)\$ END	2	
PUSH	SP_out, Clear Y, Set Carry, F=SUB, Z_in		
	Z_out, SP_in, MAR_in, Write, WMFC		
	$\mu\text{-PC}$ <= PLA(IR)\$ INTERRUPT after PUSH, JSR after PUSH	2	
	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 desti-
	μ-PC <= PLA(IR)\$ END	1	
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