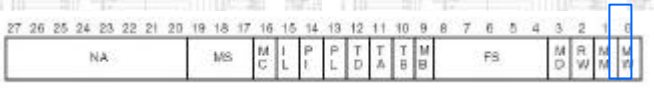
Multiple-Cycle Design

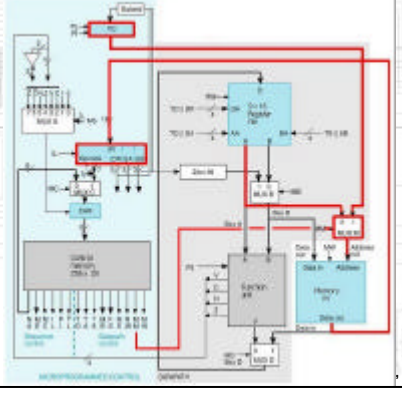
- The Multiple-Cycle Implementation demonstrates the use of a single memory for:
  - Data
  - Instruction
- This design is also used to show the implementation of more complex instructions

Memory M Address

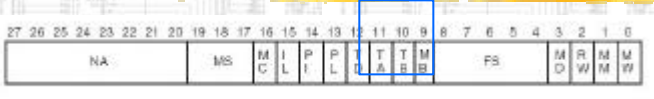


- The following address sources are used to fetch:
  - Instructions -> PC Program Counter Register (16bit)
  - Data -> Bus A (16bit)
- MUX M selects between the two address sources through the MM control signal

PC - Bus A - MM

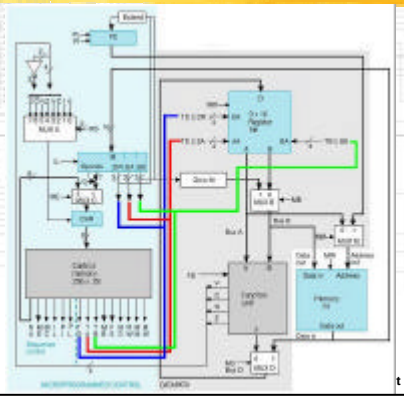


Temp Register



- Instructions are executed over multiple clock cycles
- This requires an additional register
  - R8 for temporary storage
- This register should be selected through an additional bit control signals:
  - TD, TA, TB
- These control signal are to the left of:
  - SA, SB, DR (from IR register)

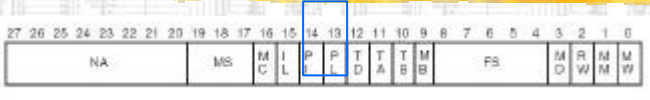
TD||DR -TA||SA - TB||SB



IR Instruction Register

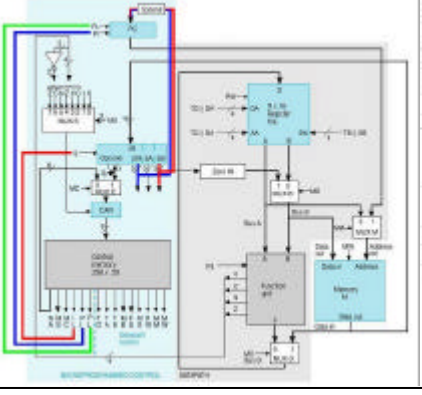
- Instructions must be held in an register during the execution of multiple micro-ops
- The IR is only loaded if an instruction is fetched from memory M
  - The IR has an load enable control signal IL
  - This signal is part of the control word

PC Program Counter Register

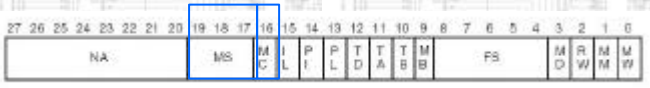


- The PC only increments if an instruction is fetched from memory M
- The control word has two bits that determine the PC modifications:
  - PI - increment enable signal
    - PC ← PC + 1
  - PL – PC load signal
    - PC ← PC + se AD

IR - IL - PC - PI - PL

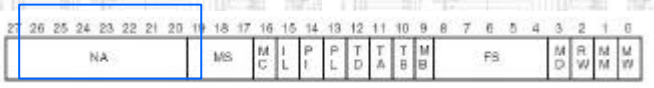


Next Address Logic



- The CAR Control Address Register selects the control word in the 256x 28 control memory
- The next logic (MUX S) determines whether CAR is incremented on loaded.
  - Controlled with MS
- The source of the loaded address is determined by MUX C
  - Selected by MC

Next Address Field

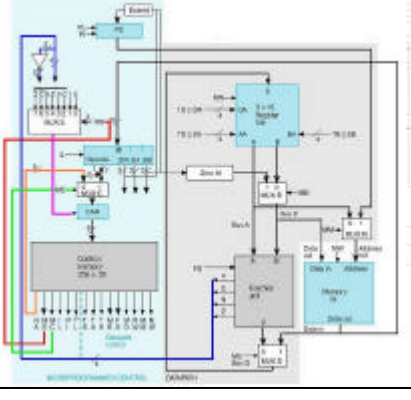


- The sources for the multiplexer can be:
  - Contents of the 8 bit NA Next Address field
  - 7 bit from the opcode field in the IR
- An opcode loaded into the CAR points to:
  - Microprogram in Control Memory
  - This program implements the instruction through the execution of micro operations
- MUX S determines whether the CAR is:
  - Incremented
  - Loaded

Sequencer Control Fields

	MS		MC		IL		PI		PL	
Action	Symbolic Notation	Code	Select	Symbolic Notation	Action	Symbolic Notation	Action	Symbolic Notation	Action	Symbolic Notation	Code
Increment CAR	CNT	000	NA	NXA	No load	NLI	No load	NLP	No load	NLP	0
Load CAR	NXT	001	Opcode	OPC	Load instr.	LDI	Increment PC	INP	Load PC	LDP	1
If C = 1, load CAR; else increment CAR	BC	010									
If V = 1, load CAR; else increment CAR	BV	011									
If Z = 1, load CAR; else increment CAR	BZ	100									
If N = 1, load CAR; else increment CAR	BN	101									
If C = 0, load CAR; else increment CAR	BNC	110									
If Z = 0, load CAR; else increment CAR	BNZ	111									

NA - MS - MC



Microprogram ASM