Part 2 – Simulator Electronic Description

1 Introduction

Part 1 concerns the installation and operation of the simulator. This part is a description of the simulator electronics.

The design implementation takes into account a possible practical implementation of the circuitry. Therefore, a mix of logic gate types is avoided where possible if the logic can be implemented simply using gates of the same type so that a practical build keeps the chip count to a minimum. For example, circuits in the following may consist of gates and inverters where a single gate of the opposite logic type would suffice.

Note that circuit modifications required for a practical build would need to include a means to view and alter RAM content and key de-bouncing.

2 Overview

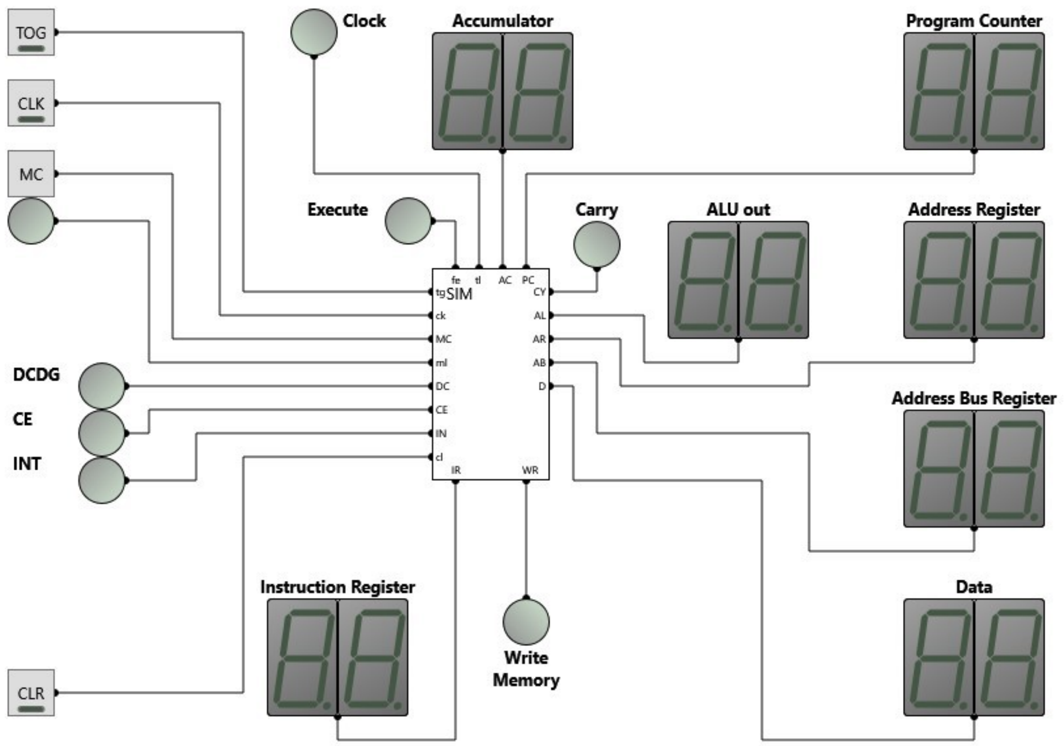


Figure 1 Simulator Main top-view

The simulator top-view Figure 1 only shows the user interface with the functional elements encapsulated in the component “SIM”. Double-clicking SIM opens the SIM circuit view Figure 2. (In the following each circuit is opened by double-clicking the circuit).

Each of the dual 7-segment displays is composed of two 7-segment display “primitives” (that is, a basic component available in the LogicCircuit tool) and ROM decoders to convert the four-bit hexadecimal data into codes which light the appropriate segments of the display.

The connecting lines between components in the circuits can consist of single or multiple wires.

The SIM circuit comprises four main elements.

- Select
- Timing
- CPU
- Register 373

“Select” controls the clocking of the simulator using the controls described in Book 3 Part 1. The tool clock is introduced at this point. The clock pulse(s) appear at the output pin “clock” according to the status of CLR (on pin “clear”) and the selected input key (“tg”, “CLK” or “MC”). The timing input pin “dcdg” derived from the named timing pulse provides the control for the MC key operation.

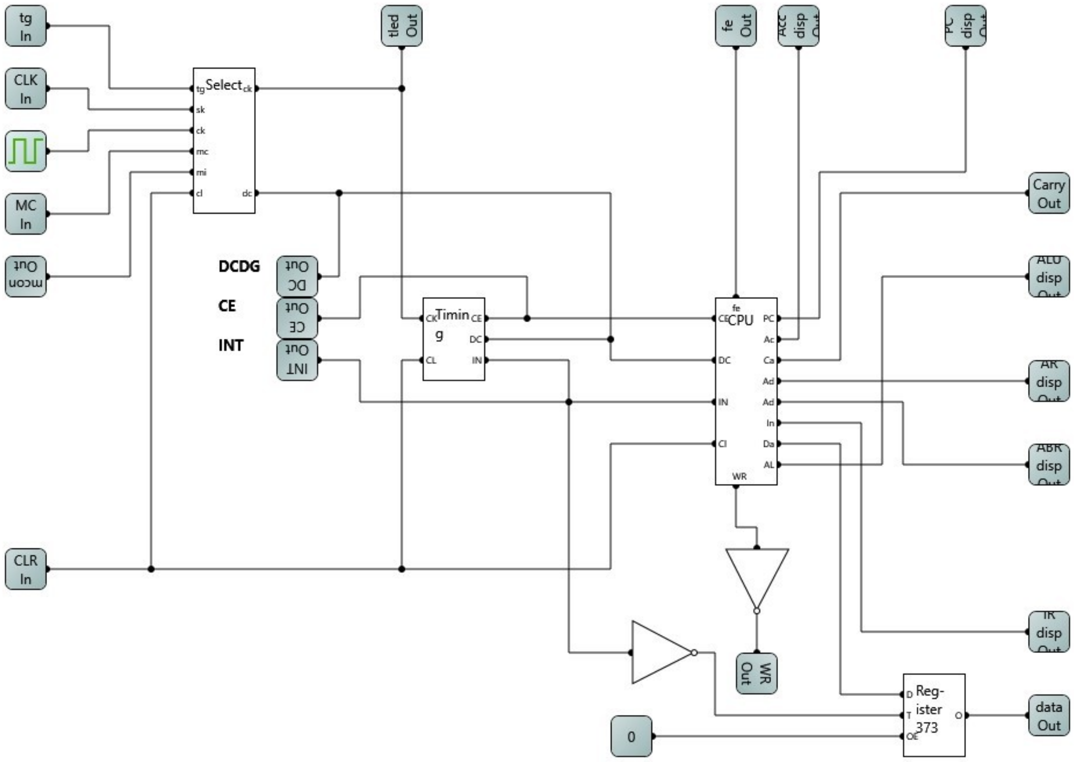


Figure 2 SIM circuit

“Timing” generates the three timing pulses used by the simulator electronics. Timing would more naturally appear within the CPU, if the circuit was fabricated on a chip, but appears here separately for simplicity. The three outputs are derived from CLR and the clock.

“CPU” contains the main circuitry of the simulator processor. The pins include the timing pulses, CLR and display outputs. The CPU includes the RAM memory (256 bytes). In practice the memory may be separate from the CPU in a fabricated form but is included within the CPU here for simplicity.

“Register 373” is a commercial ic device and is used to latch the data bus, which is released at the end of a machine cycle and therefore contains no data (and hence would not display anything at the end of the machine cycle). The data is latched using the timing pulse INT.

In the circuits many devices are referred to using numeric references (e.g. 373, 7474). The numbered devices are commercially available and the characteristics for each are derived from the corresponding commercial specification.

3 Timing

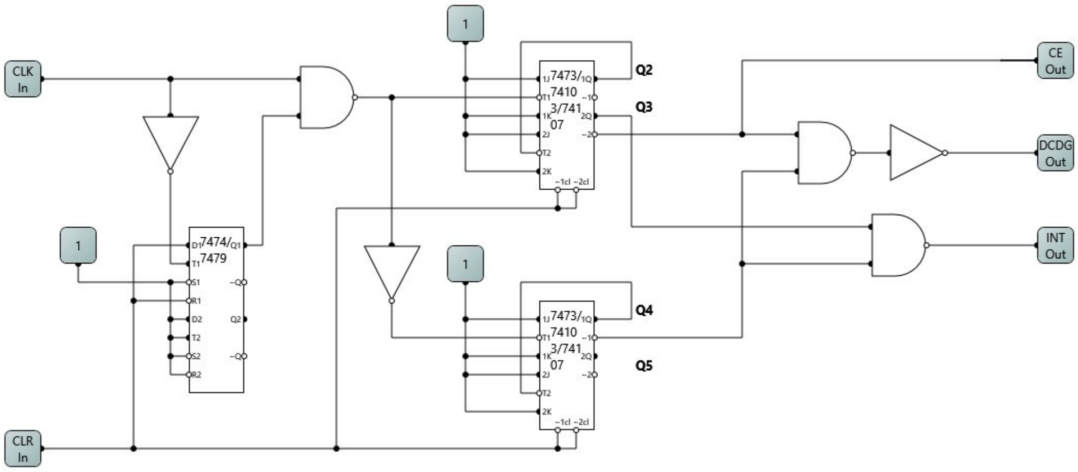


Figure 3 Timing circuit

The Timing circuit Figure 3 consists of three parts

- An edge detector and clock enable to the left
- Two anti-phase clock dividers in the middle
- A set of decoders generating the timing pulses to the right

When CLR enables the simulator electronics (i.e. clear is set to one) the state and phase of the clock is indeterminate. The edge detector circuit ensures that the timing circuits always begin with a clean and full-phase clock pulse.

The two 7473 devices are JK flip-flops arranged to divide the applied clock pulses. On power-up CLR holds the Q outputs on zero. The two T1 timing pulses are anti-phase and the Q outputs when CLR is set to one and the clock is running for each is shown in Figure 4.

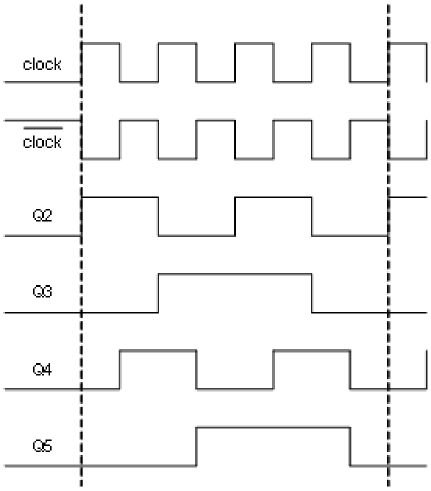


Figure 4 JK Flip-flop output pulses

In the figure, Q2/Q3 change on the rising edge of clock and Q4/Q5 on the rising edge of clockbar. Each pair only changes when the other pair is stable and unchanging.

The required timing signals over a machine cycle (from Part 1) are as shown in Figure 5.

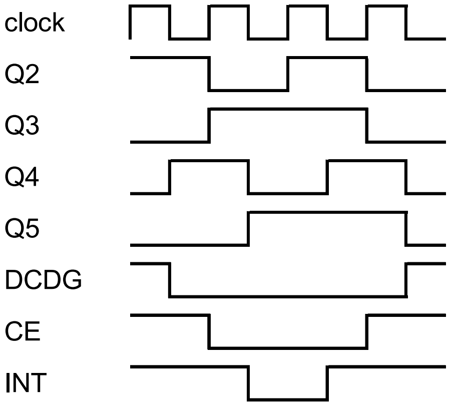


Figure 5 Flip-flop outputs and required timing pulses

To derive the pulses the outputs Q2/Q3 are gated with Q4/Q5. The anti-phase outputs ensure there are no timing hazards with the pulses since at no time do the input pulses change at the same time.

Analysis of the required outputs results in the following functions.

$$\mathrm{DCDG} = \overline{\mathrm{Q3}} \;.\; \overline{\mathrm{Q4}}$$

$$\mathrm{CE} = \overline{\mathrm{Q3}}$$

$$\mathrm{INT} = \overline{\mathrm{Q3} \;.\; \overline{\mathrm{Q4}}}$$

The Boolean functions lead directly to the decode circuits on the right of the circuit.

4 Select

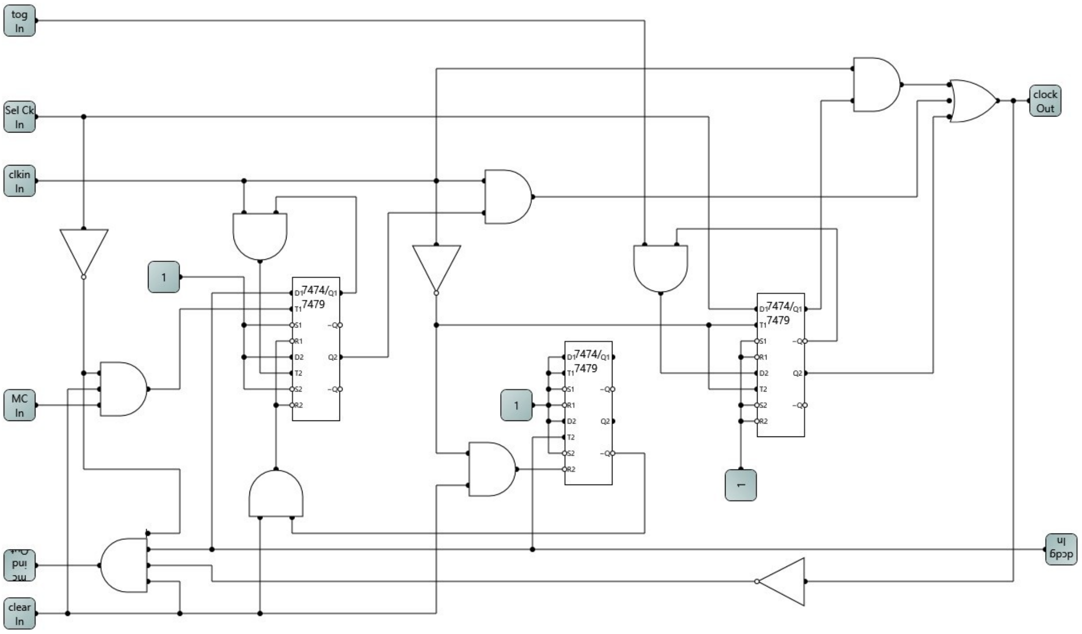


Figure 6 Clock Select circuit

This Select circuit Figure 6 determines which of the clocking options is applied to the Timing circuit. It is controlled according to the settings of the switches on the inputs “tog”, “Sel Ck” and “clear”. The pulses appear (when on) on the output “clock”. The LogicCircuit tool clock is applied on the input “clkin”.

Most of the circuit, the inputs “MC”, “dcdg” and output “mc ind” are concerned with the MC clocking operation.

The logic gates at the top right of the circuit act upon the signal from the TOG and CLK keys controlling the output “clock”. The CLK key on input Sel Ck selects either “tog” or “clkin” and the actions of both keys are synchronised to the tool clock (in a manner similar to the edge detector in the Timing circuit) by the 7474 flip-flop on the right of the circuit. The circuit prevents glitches which can occur with an asynchronous arrangement.

When the Sel Ck input is high the output “mc ind” is forced off (if on) and the MC key is disabled.

When the Sel Ck input is low the MC and TOG keys are enabled (TOG via the 7474 latch and clkin). When the output “clock” is high the output “mc ind” is forced off (via the inverter). This is a very simple indication that MC should not be operated when a machine cycle has started. The circuit does not prevent novel or unusual operation of the MC and TOG keys together but there are no detrimental effects.

The following operation of the MC key occurs when the CLK and TOG keys are not selected (i.e. normal operation of MC).

With the CLR key not selected (as on power-up) and the input “clear” low the MC key is disabled, the “mc ind” output forced low, the Q outputs of the left D flip-flops 7474 are forced low and the Qbar output of the middle 7474 is forced high.

Clocking the CPU may begin when CLR is selected and “clear” goes high. All inputs on the four-input AND are high (“clock” is set low and “dcdg” set high on power-up) and “mc ind” goes high, lighting the LED. The MC key is enabled through the three-input AND.

The forced resets on the left 7474 are removed by setting them high through the AND of “clear” and Qbar of the middle 7474. The forced reset of this 7474 is continually applied whenever “clkin” goes high. The output Qbar is set high anyway and this circuit has no further operation until after MC is pressed.

The input trigger of the first D flip-flop in the left 7474 is activated whenever the MC key is enabled and pressed. Note this is whenever the CLR key is selected and the CLK key is not selected. The input D is transferred to the Q1 output on every positive edge of the trigger. The signal on D is determined by “dcdg”, so the Q1 output is high when “dcdg” is high.

If “dcdg” is low (i.e. machine cycle has started) when MC is pressed then the Q1 output is set low and no further activity occurs. That is, pressing MC has no effect (and the LED is off).

If “dcdg” is high when MC is pressed then Q1 is set high and triggers the second flip-flop when the clock is high. Q2 is set high which enables “clkin” to appear at “clock”. The state is stable and clock pulses continue to appear at the “clock” output.

The falling edge of the fourth clock cycle results in DCDG going high from the Timing circuit. R2 of the middle 7474 is set high (i.e. becomes inactive). In fact, the transition time for DCDG is much longer than the signal to R2 because the clock must transition the Timing circuit. Therefore, R2 is high and stable before the trigger arrives on T2. Q2bar is set low.

The low from Q2bar is fed back to reset both the flip-flops on the left 7474. “clkin” is disconnected from “clock” and the trigger for the MC key is reset. The “mc ind” output goes high and the LED lit. The next rising edge of the clock resets Q2bar high on the middle 7474. The Select circuit has returned to its starting state.

5 CPU

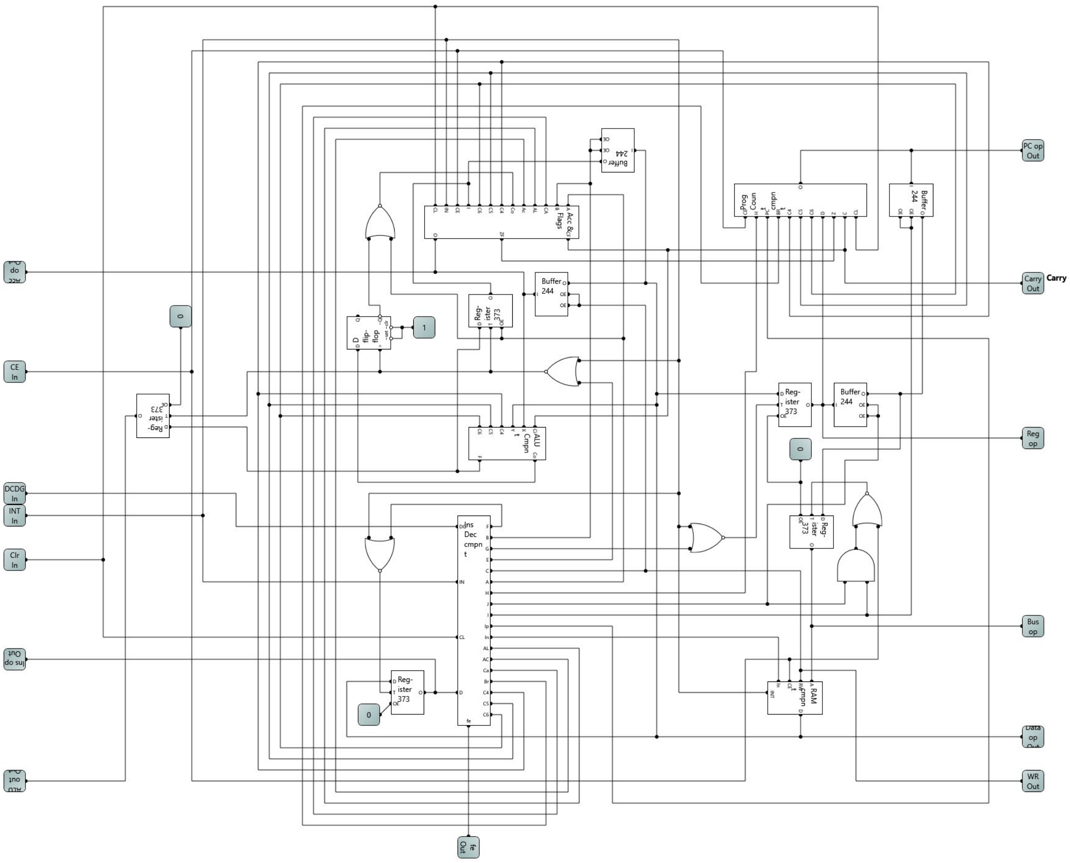


Figure 7 CPU circuit

The CPU circuit Figure 7 contains the six registers making up the simple processor along with the Gates (A to J) and Memory. The registers are shown in Figure 8.

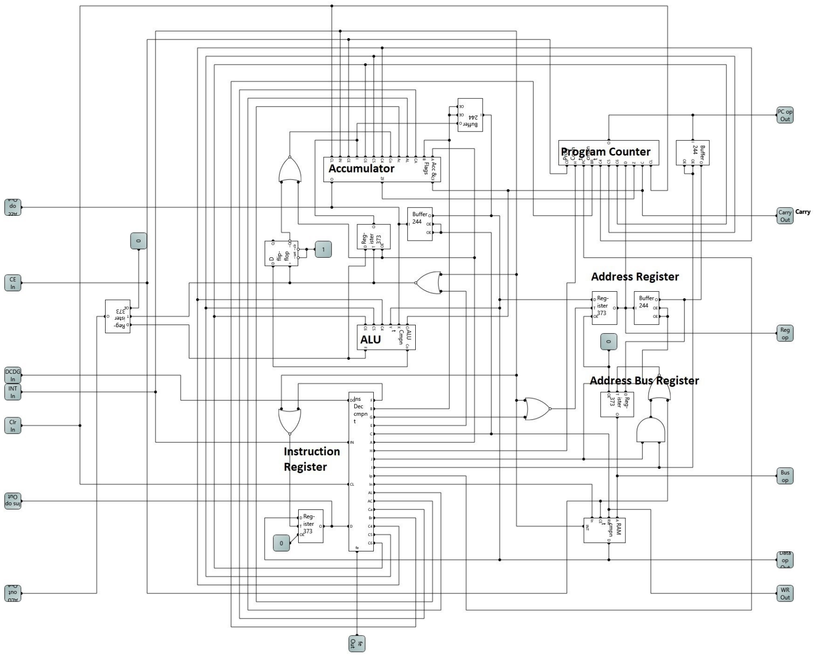


Figure 8 Register identification in the CPU circuit

In the figure the Accumulator, Program Counter, ALU and Instruction Register contain further circuitry described in the following. The Address Register and Address Bus Register are commercial eight-bit latches (74373). The logic in the figure between the registers is all controlled by the output of the Instruction Register (IR). A detail view of the IR input/output is shown in Figure 9.

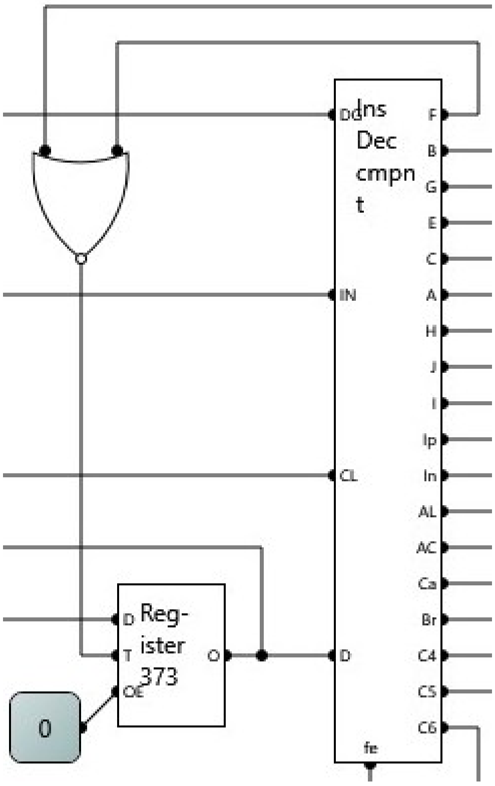


Figure 9 Instruction Register input/output

The timing pulse inputs to the IR are DCDG and INT. Clear (Reset) is also applied. The Register 373 holds the machine code value of the Instruction when F is low (i.e. the Gate output signals A to J are active low) during the Fetch cycle and INT is applied to the NOR gate.

The decoder outputs consist of the Gate controls A to J (D omitted – see Book 1), controllers for incrementing the program counter (Ip) and internal operation (In), a selector for each of the ALU, Accumulator, Branch and Carry function processing electronics and C4 to C6 (Control 4 to 6 in Book 1 Part 2) which set the detail processing for each function. See Book 1 Part 2. The selectors and C4 to C6 connect directly to the required Registers.

The following figures illustrate the operation of the Gates during certain machine cycles. The Instruction Register decoding is described in section 6.

5.1 Fetch

With the IR in the Fetch state the IR outputs are set active as shown by the bold lines in Figure 10. Gate F and I are active along with Increment Program Counter during DCDG active. Gate I enables the Program Counter output to be placed in the Address Bus Register and on to RAM by the timing pulse CE. Subsequently the data addressed in RAM (in read mode) flows through to the IR latch when INT becomes active (i.e. low). The data is latched when INT returns high. RAM is enabled by the timing pulse CE.

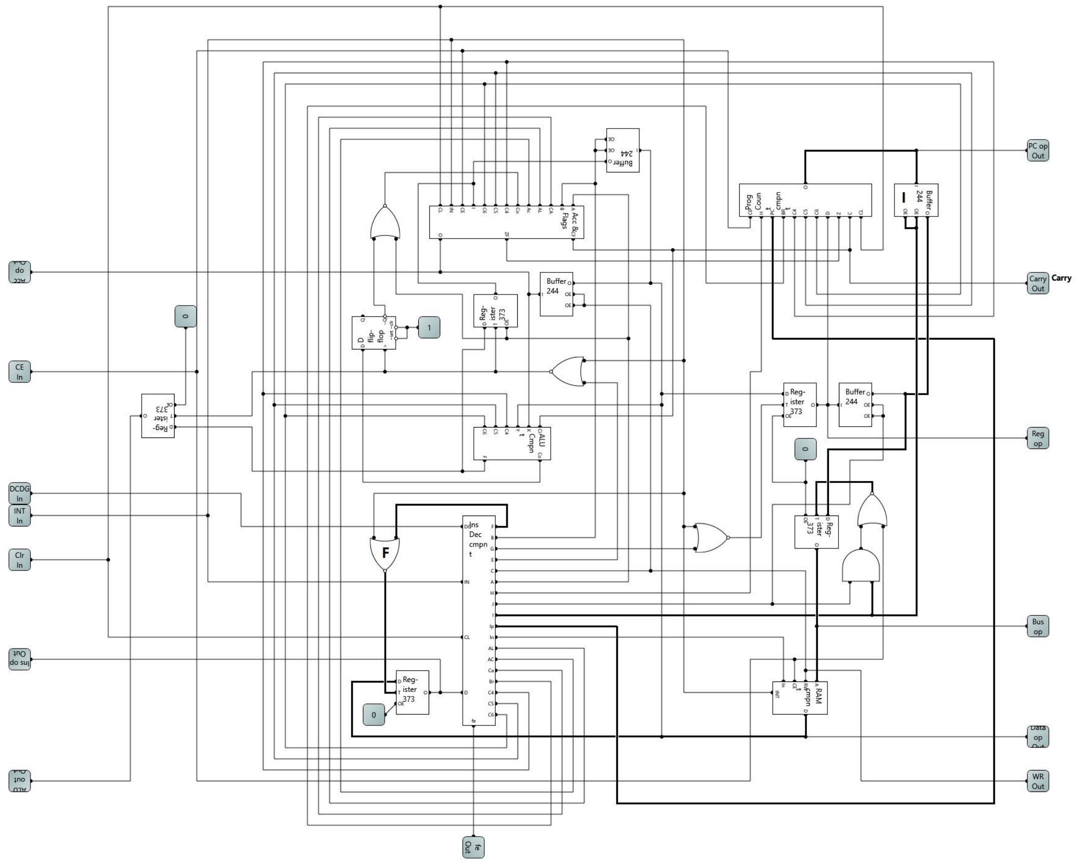


Figure 10 Active control/data lines during Fetch cycle

The active line Ip to the Program Counter determines that the PC is incremented when CE becomes inactive (i.e. goes high).

The decoded output of the IR appears when the Execute state is invoked on the completion of Fetch (section 6).

5.2 Load Immediate

LDI consists of a single execute cycle. The Gates enabled are I and B. The increment pc line is also active. See Figure 11.

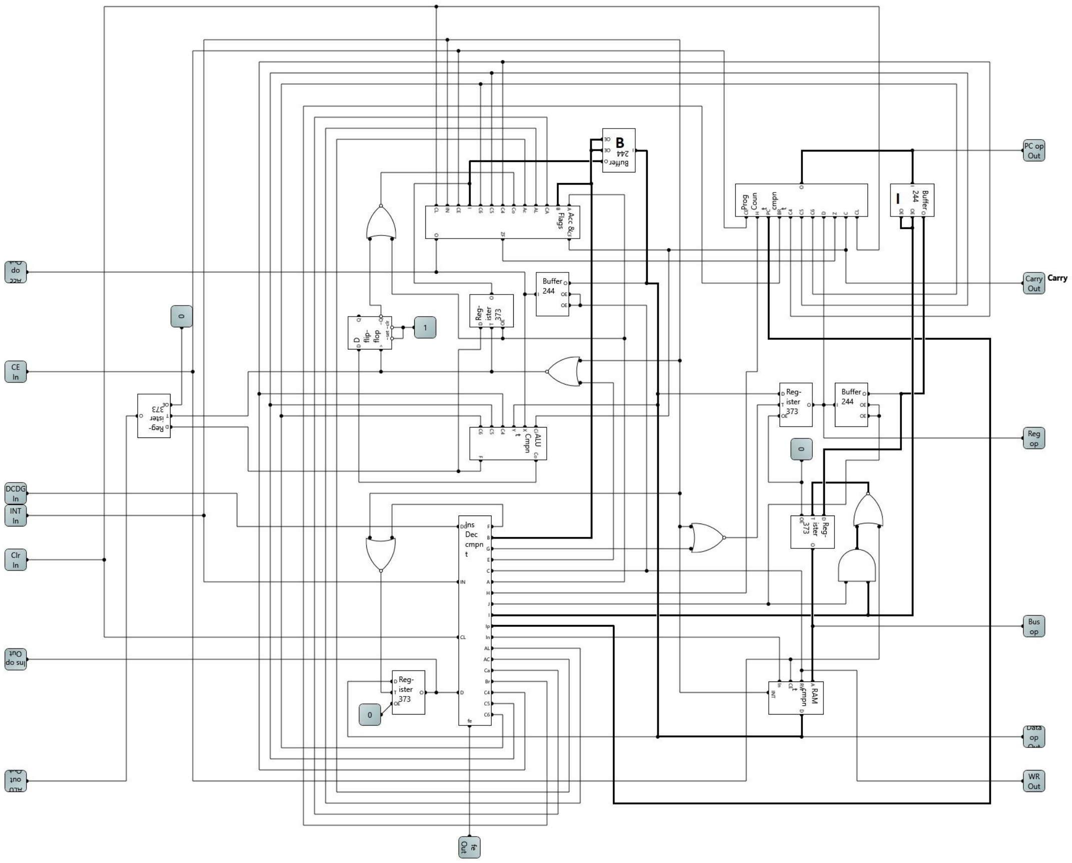


Figure 11 Active control/data lines during LDI Execute cycle

5.3 Load Accumulator

The Address Register is loaded during the first execute cycle of this instruction and the machine cycle is common to many other instructions using addressed memory (i.e. Store, ALU and Branch). Gates I and G are opened. Increment PC is active. See Figure 12.

During the second cycle the Gates J and B are opened. The PC is not incremented (Ip is not active). See Figure 13.

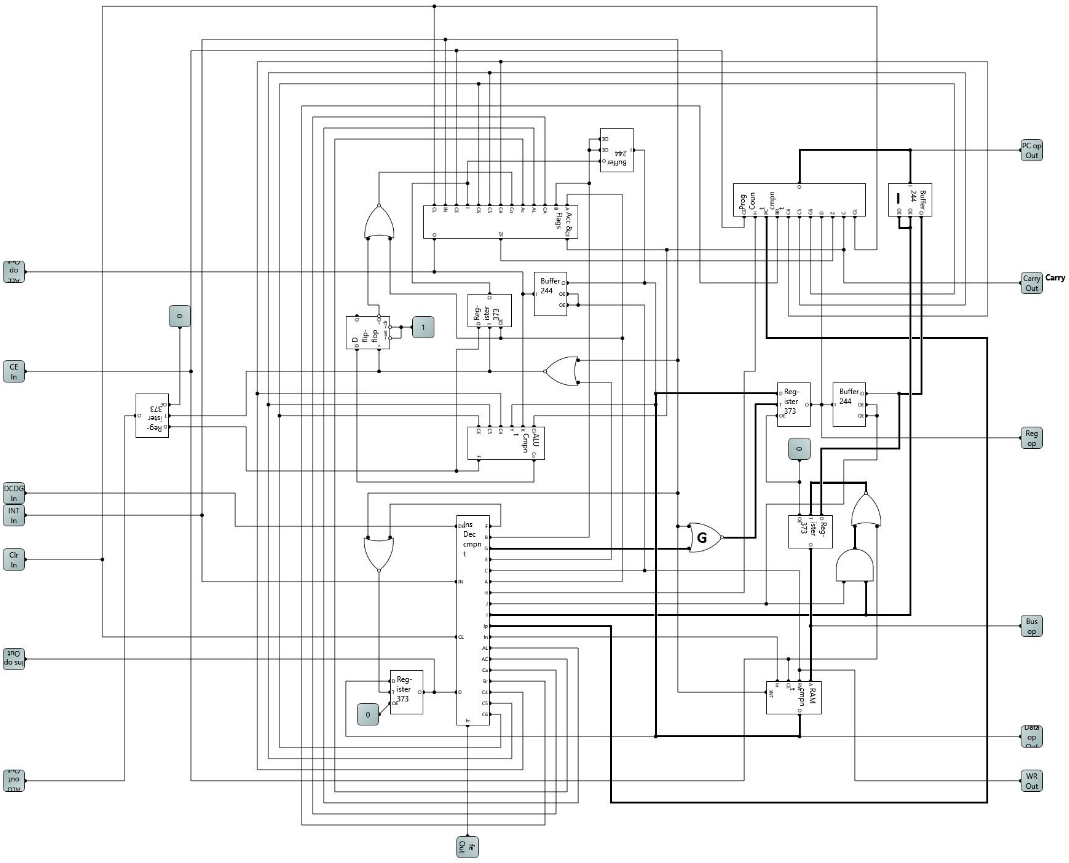


Figure 12 Active control/data lines during a load of the Address Register

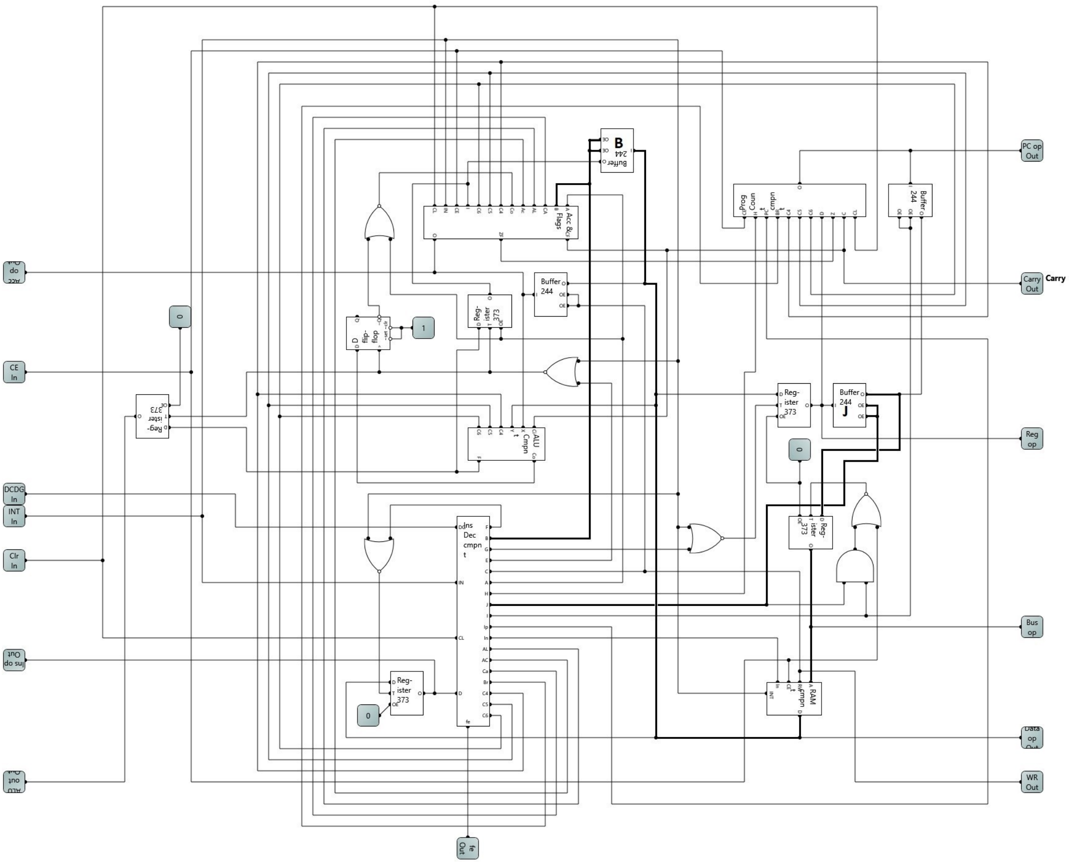


Figure 13 Active control/data lines during second LDA Execute cycle

5.4 Store Accumulator

After the address is stored in the Address Register as seen in section 5.3, the Accumulator data is copied to memory by opening Gates J and C. C becoming active places the RAM into the write mode. The data is latched in memory when INT becomes inactive.

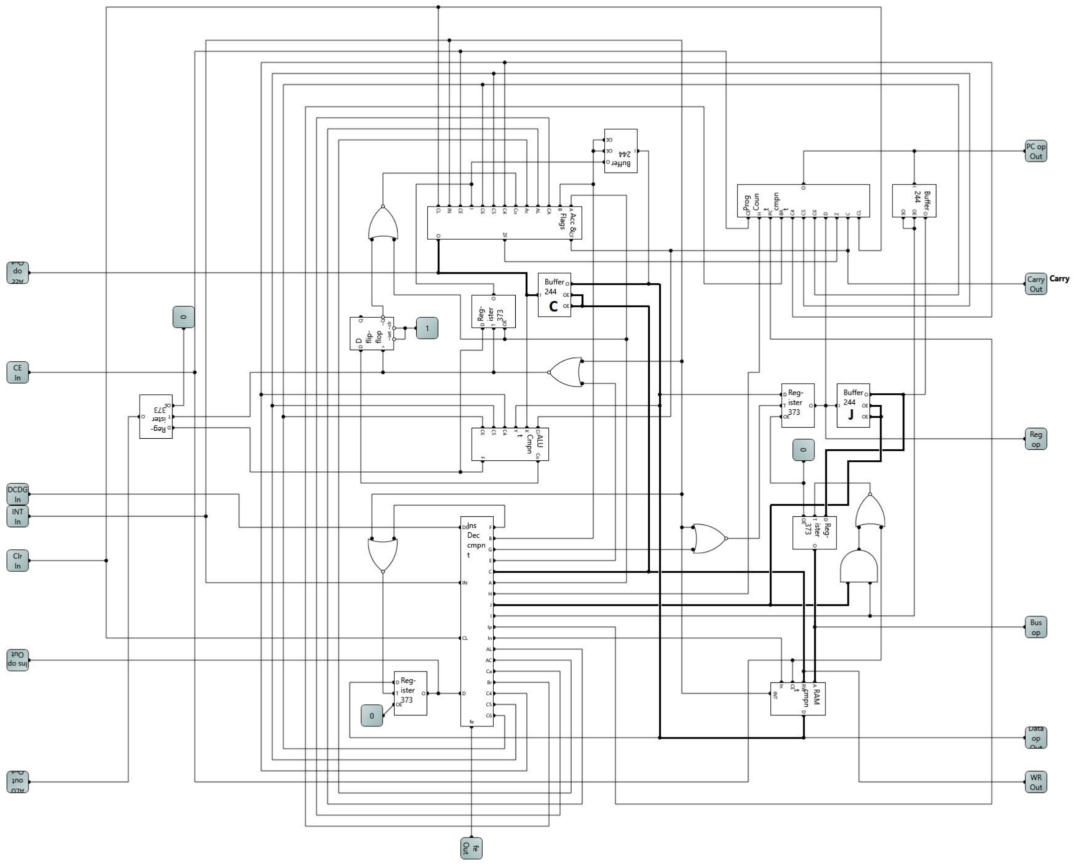


Figure 14 Active control/data lines during second STA Execute cycle

5.5 ALU Operations

ALU instructions operating with addressed data are four machine cycles. The Address Register is set-up as seen before (Figure 12). The calculation is executed in the third cycle by opening Gates J and E as shown in Figure 15.

The ALU logic is connected to the data bus and to the Accumulator. This means that ALU calculation occurs at all times that the data appears on either. However, only the data passed during Gate E becoming active is captured in the Register 373 and the flip-flop associated with the ALU (so the simulator implementation is slightly different to the architecture described in Book 1 and in the emulator Book 2). The D flip-flop captures the Carry result (but is only used to update the Carry flag during arithmetic instructions).

The Register 373 to the far left in the figure also captures the relevant ALU output. This is for display purposes only and does not figure in the processor computation. Its output is permanently enabled.

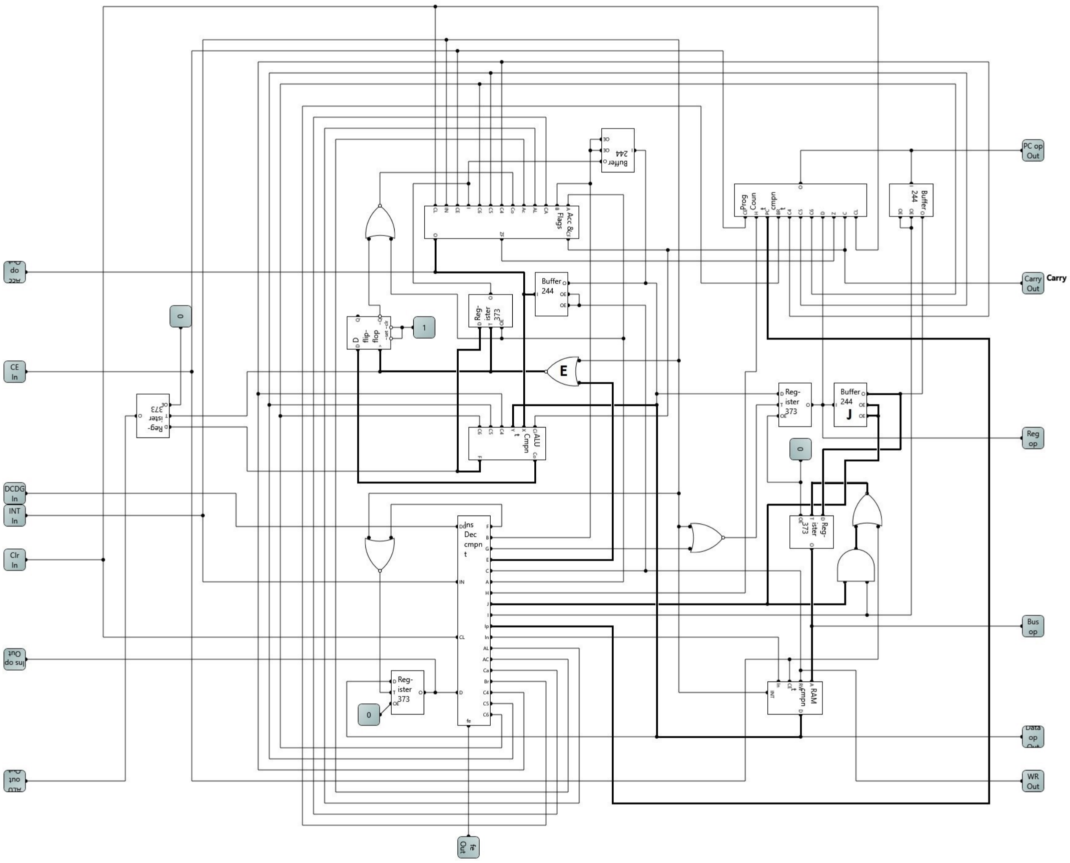


Figure 15 Active control/data lines during ALU calculation Execute cycle

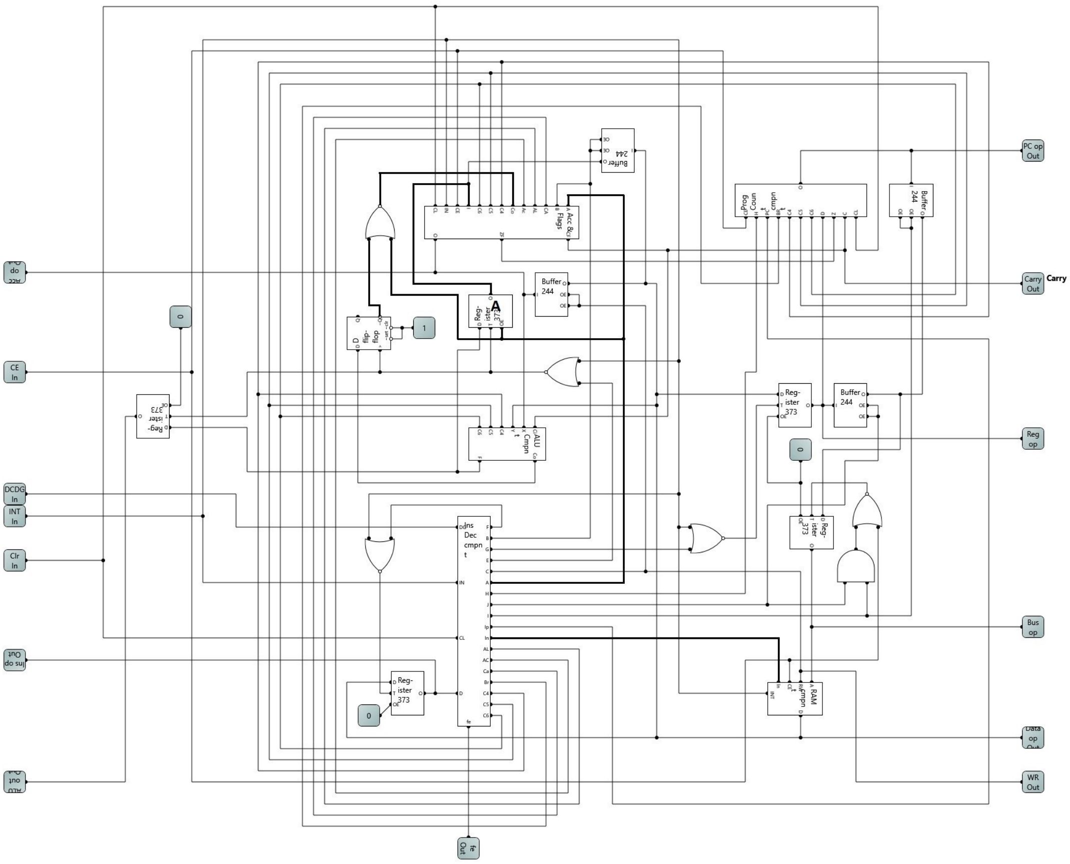


Figure 16 Active control/data lines during ALU result Execute cycle

ALU instructions include immediate data address types. The Gate sequence replaces the Address Register set-up with the Program Counter in a manner similar to the LDI example.

In the final cycle Gate A is opened and the data captured in the Accumulator (Figure 16). The Internal Operation line is active and the RAM output is disabled.

5.6 Branching

The final cycle during branching sees the Carry/Zero flags held in the Accumulator and the output of the Address register processed in the Program Counter according to the values C4, C5 and C6. Figure 17 shows both the Carry and Zero flag lines as active but in practice only one (or none if branch is unconditional) is used.

Branching is activated simply by enabling the timing signal INT within the Program Counter (section 8) when H is active.

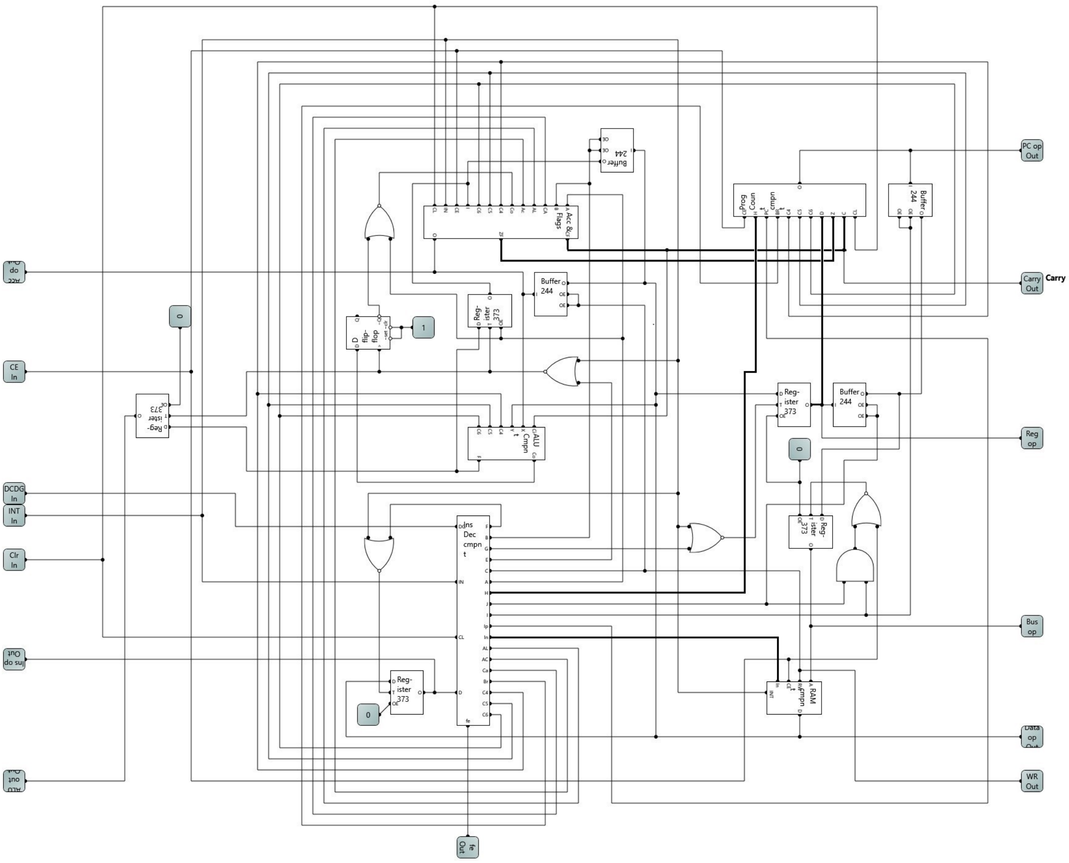


Figure 17 Active control/data lines during second Branch Execute cycle

6 Instruction Decoding

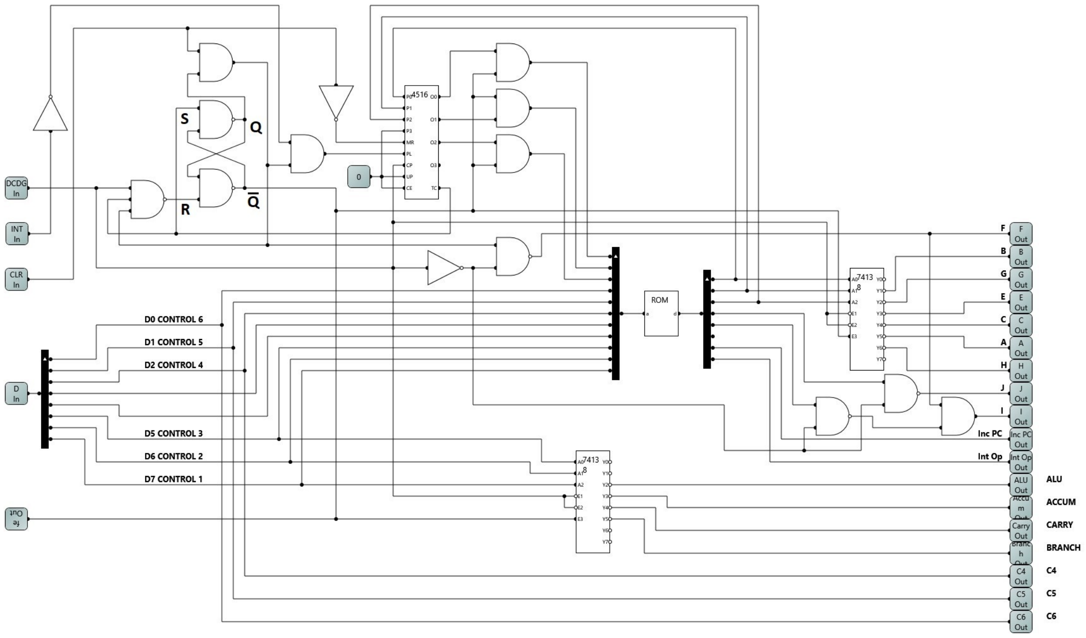


Figure 18 Instruction Register Decoder circuit

Book 1 Part 2 describes the principles underlying the decoder design shown in Figure 18. The core decoding is achieved using a ROM which is addressed directly by the Instruction Code. The high eight bits of an 11-bit address are formed by the Instruction Code whilst a counter (4516) addresses one of the eight bytes in the addressed block.

On power-up CLR is held low leading to the following

- The 4516 is held in reset with its Qn outputs and TC low.
- The SR flip-flop set S is held low by TC: Q is high and Qbar low whilst the flip-flop reset R is high.
- Qbar disables the 74138 decoders and the three AND gates hold the ROM low three bits at zero (Gates A, B, C, E, G, H and selectors inactive).
- The output of the AND on Q is held low holding the NAND generating the Gate F output high (i.e. F inactive).
- Similarly Gates I and J are inactive (DCDG is high).

CLR set high alters the state as follows

- The AND output fed from SR flip-flop Q goes high.
- The 4516 reset is released and outputs are unchanged.
- The Gate F NAND active input becomes DCDG, which is currently high and hence Gate F remains inactive.
- INT becomes the active input of the AND feeding the PL input to the 4516.
- The 74138 decoders remain disabled by SR flip-flop Qbar and DCDG high.

The states are stable and unchanging until clocking begins.

DCDG goes low.

- The SR flip-flop state is unchanged. The 74138 decoders remain disabled.
- The Gate F NAND output goes low and F becomes active. The AND gate forces I active and the Fetch state on the processor begins.
- The PC output feeds through the Address Bus Register to the RAM.

CE goes low.

- The RAM data (Instruction) appears.

INT goes low.

- The RAM data is copied into the Instruction Register 373 (visible on the CPU figure) feeding the D input and hence the ROM inputs.
- The low three bits of ROM are zero. The addressed data appears at the ROM output.
- The Increment PC output becomes active (set in ROM).
- PL input to the 4516 goes high and the low three bits of the ROM output are copied into the low three bits of the counter.
- The 4516 TC output goes high removing the active set S on the SR flip-flop.
- The three input NAND output is now only held high by DCDG low.

INT goes high.

- Instruction is latched into the IR 373 (see CPU figure).
- Count is latched into the 4516.

CE going high has no impact on decoding during Fetch.

DCDG goes high.

- The three-input NAND output goes low and forces a reset on the SR flip-flop. The Q and following AND outputs are forced low.
- The three-input NAND goes high and the reset is no longer active (the active state is transitory).
- The PL AND output goes low and the INT input to the 4516 PL becomes inactive.
- DCDG going high decrements the counter.
- DCDG high holds the 74138 decoders disabled.
- SR flip-flop Qbar output is high and the counter output AND gates are enabled. The low three bits of the ROM are updated. New ROM output is made available.
- The Gate F NAND output goes high and the F and I outputs become inactive.

The counter is loaded and active with the SR flip-flop Qbar output high and the Instruction decoder is now in the Execute state (fe output is high).

DCDG goes low.

- No impact on SR flip-flop.
- 74138s, I and J outputs become available for activation according to the ROM output. (ROM output available since DCDG went high in last cycle).

CE and INT have no effect on IR Decoder during Execute.

DCDG goes high.

- Gate outputs disabled
- No impact on SR flip-flop
- Counter decremented. New ROM output available.

The Execute states repeat until the counter decrement sets the TC output low (when DCDG goes high) indicating the count is zero and no more execute states remain to be processed. TC going low leads to the following

- The SR flip-flop set S is forced low and Q goes high.
- Qbar is set low disabling the counter output AND gates.
- The 74138 decoders are disabled.
- The counter PL input AND is enabled and INT becomes the active input.
- The Gate F NAND is set so that DCDG is the active input again. The Fetch state is reinstated (fe output is low).

And the next instruction machine cycle begins.

7 Accumulator

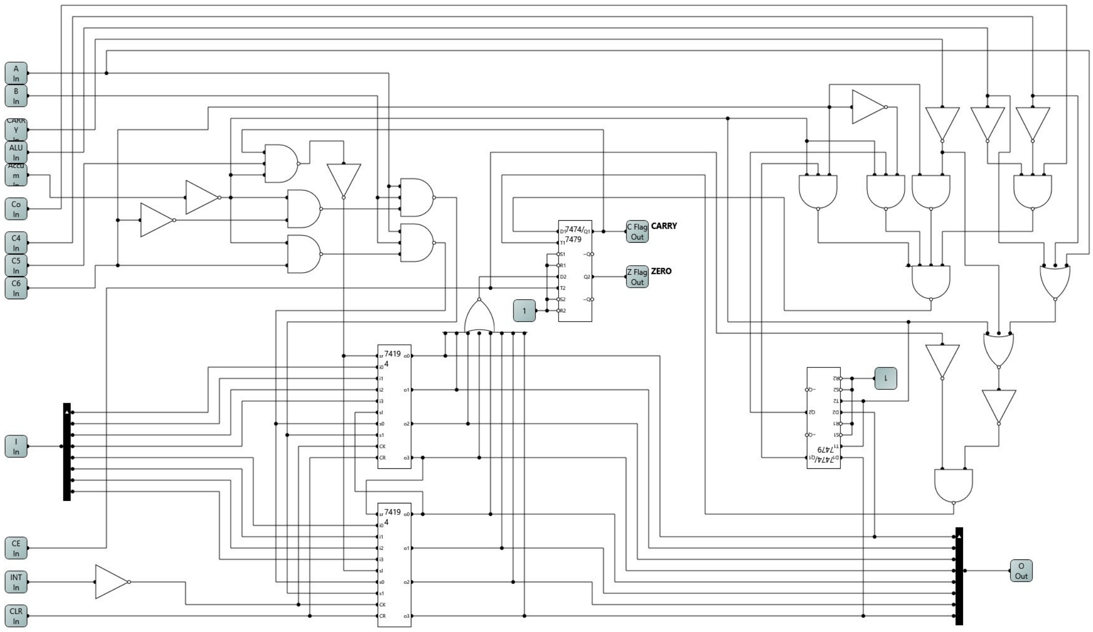


Figure 19 Accumulator and Flags circuit

This circuit in Figure 19 handles the Accumulator functionality and flags processing (Carry and Zero). The flags are held in the 7474 D flip-flops at the centre of the circuit. The logic to the left of the flags is concerned with the Accumulator and to the right the Carry flag. The inputs A and B relate to Gates A and B respectively.

7.1 Carry Flag Circuit

The Carry flag is impacted by a variety of processor functions.

- Explicitly set/reset by Instruction Code
- ALU result
- Accumulator shift/rotate

Hence the decoding is relatively complex.

Two sets of decoding decide

- when the Carry flip-flop is to be triggered
- what the Carry value is to be.

The D flip-flop trigger for Carry is set by CE going high provided the three-input NOR gate to the bottom right of the figure is not held off by a high output from the preceding three-input NOR. The Truth Table for the preceding NOR is as shown in Table 1. Note that the inputs shown are the only occurring values which feed the circuit and not all theoretically possible values. (X indicates “don’t care”). The inputs are active low.

A	CARRY	ALU	ACCUM	C4	Nor Output	Outcome (Nor output)
1	0	1	1	X	0	Carry selected enables setting Carry
0	1	0	1	0	0	ALU selected with C4 low and Gate A open enables setting Carry. Arithmetic function
0	1	0	1	1	1	ALU selected with C4 high and Gate A open disables setting Carry. Logical function
1	1	0	1	X	1	ALU selected and Gate A closed disables setting Carry
1	1	1	0	X	0	Accum selected enables setting Carry
1	1	1	1	X	1	No feature selected disables setting Carry

Table 1 Carry Flag trigger logic

The data set into the Carry Flag is determined for each of the selected functions (Carry, ALU, Accumulator) using Co (the captured Carry output of the ALU), C4, C6, b0 and b7 as shown in Table 2.

CARRY	ALU	ACCUM	C4	C6	CF	Outcome (CF)
1	1	1	X	X	0	Carry flag not updated
0	1	1	X	0	0	Carry flag instruction. Carry flag set to zero
0	1	1	X	1	1	Carry flag instruction. Carry flag set to one
1	0	1	0	X	Co	ALU Arithmetic mode. Carry flag set as per Co
1	0	1	1	X	0	ALU Logic mode. Carry flag not updated
1	1	0	X	0	b0	Accum mode and Carry flag set to b0
1	1	0	X	1	b7	Accum mode and Carry flag set to b7

Table 2 Carry value determination

The Accumulator Instruction Codes (Shift and Rotate) update the Carry flag with zero, b0 or b7 depending upon the type and direction of bit movement. For rotation the Carry bit is moved into b0 or b7 of the Accumulator (section 7.2) and the existing b0 and b7 moved into the Carry flag.

The existing values of b0 and b7 are buffered in the 7474 D flip-flops on the right of the circuit. The values are captured on each selection of the Accum function. The outputs are decoded according to the Table and clocked into the Carry Flag flip-flop by CE going high (i.e. after the register shift).

7.2 Accumulator Circuit

The Accumulator data is held in the two 74194 shift register circuits wired as an eight-bit shift register. The circuits include reset functions connected to CLR. The circuits are clocked for every machine cycle by INT. The activity in the 74194 depends upon the status of the control inputs S0 and S1 derived from the outputs of the two three-input NANDs.

The MSB in the circuit (i.e. b7) is the bottom data line in the figure, with the LSB (b0) at the top. The activity is determined as shown in Table 3.

Inputs				Outputs		
A	B	Accum	C6	S1	S0	Notes
1	1	1	X	0	0	Shift Register outputs do not change
0	1	X	X	1	1	Data on input lines clocked into registers when Gate A open
1	0	X	X	1	1	Data on input lines clocked into registers when Gate B open
1	1	0	1	0	1	Left shift/rotate
1	1	0	0	1	0	Right shift/rotate

Table 3 74194 Shift Register function

In shift/rotate mode the LSB/MSB is set according to the output of the three-input AND (consisting of a NAND and a NOT). C5 discriminates between shift and rotate. For shift (C5 is zero) the output is zero. For rotate (C5 is one) the output is determined by the Carry Flag.

Bit shifting occurs on INT which is after b0 and b7 are captured (section 7.1) and before CE updates the Carry flag.

7.3 Zero Flag Circuit

The Zero Flag is set on every CE. The output from the accumulator (shift registers) is connected to an eight-input NOR which feeds the D input of the Flags 7474.

8 Program Counter

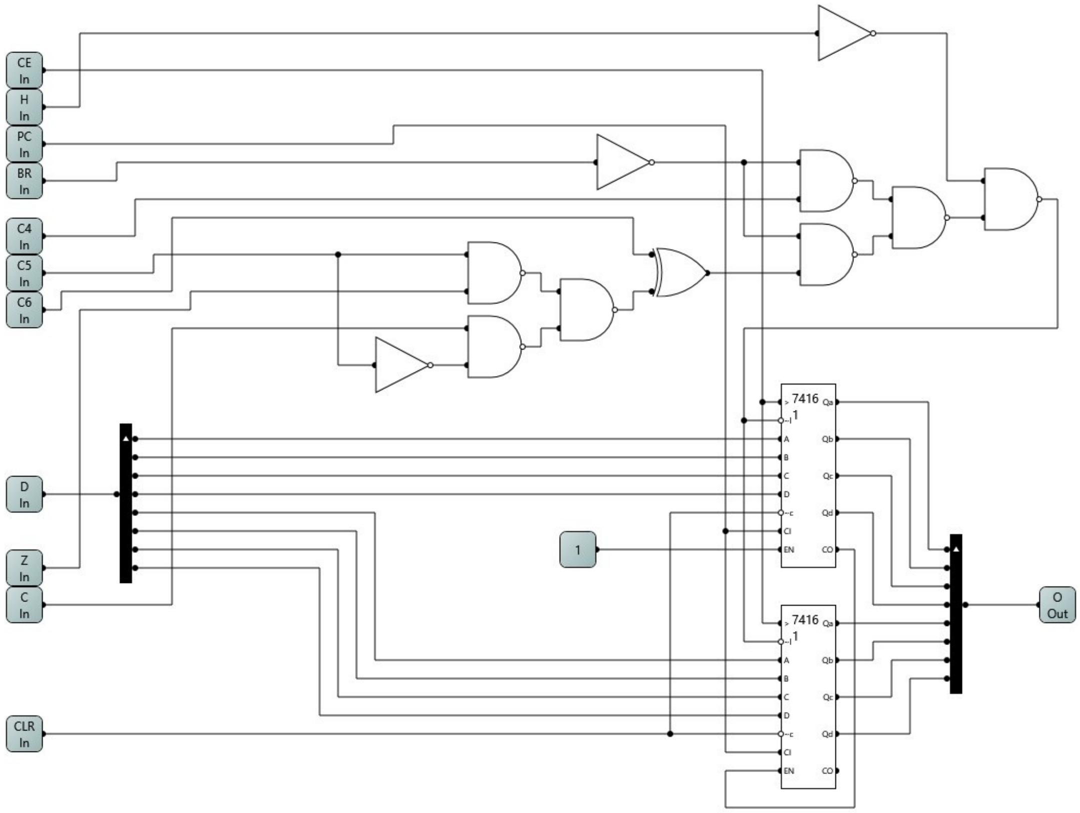


Figure 20 Program Counter circuit

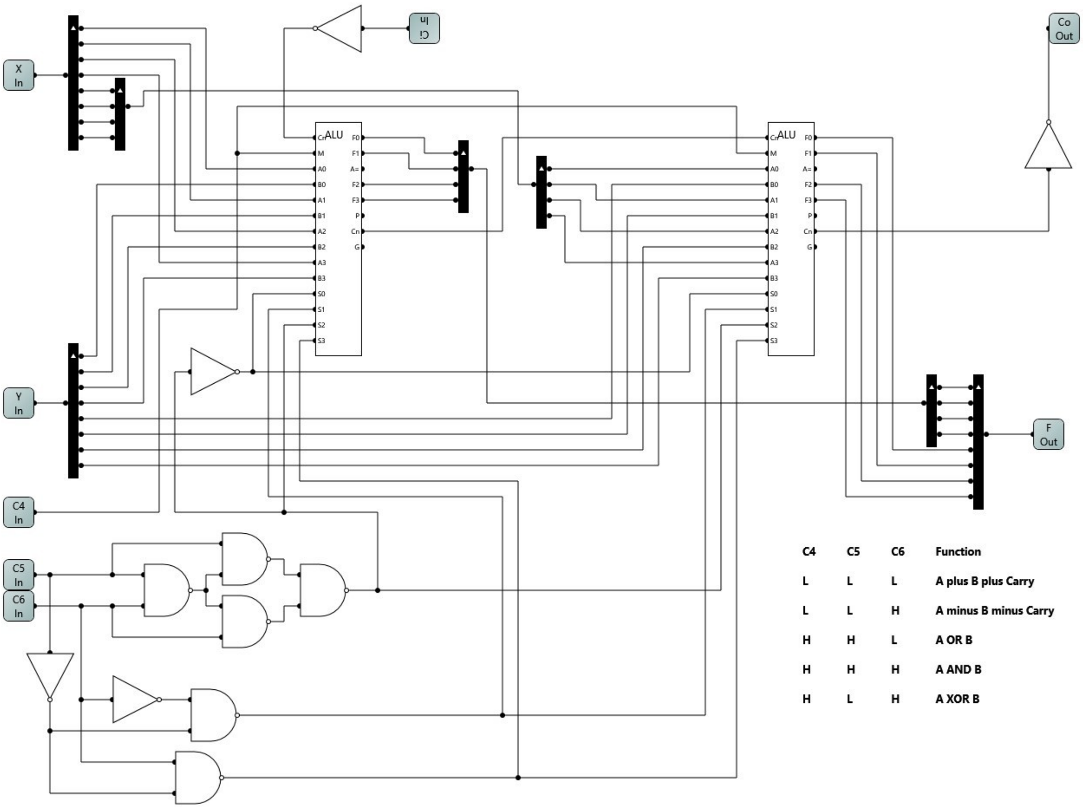
The Program Counter Figure 20 consists of two four-bit settable counters (74161) wired as an eight-bit synchronous counter and two decoder circuits controlling the count and load features. The counters are reset through CLR to 00 (hex).

The count is controlled by CE when activated by the selector PC from the IR. The Program Counter is incremented on the low-to-high edge of the pulse.

The preset count feature is used to load the content of the Address Register, which is present on the D input. The controlling pulse is activated by the selector BR. The selector enables the two NAND gates connecting C4 and the XOR gate. A high on either C4 or the XOR output leads to a load.

C4 is set high by the unconditional branch instruction (BRA). For the conditional branches C4 is low and the increment depends upon the XOR. This is fed from a decoder consisting of C5, C6, Z and C. C6 acts as an inverter (set high the XOR inverts the other input). C5 switches control between C and Z. Depending upon the required conditions the counters may or may not be loaded.

9 ALU



C4	C5	C6	Function
L	L	L	A plus B plus Carry
L	L	H	A minus B minus Carry
H	H	L	A OR B
H	H	H	A AND B
H	L	H	A XOR B

Figure 21 ALU circuit

The ALU Figure 21 consists of two four-bit ALU integrated circuits (74181) arranged as an eight-bit ALU. The ALU function is determined by C4, C5 and C6 as shown in the figure.

The ALU inputs (X and Y) are connected directly to the Accumulator and Data Bus with the output appearing at F and Co. The output is captured when Gate E is active. Note that the simulator implementation differs slightly in this respect from the processor described in Book 1 and the emulator Book 2.

Although other values for C4, C5 and C6 appear on the ALU the circuits always produce some defined output (as per the data sheet for the type) but this design ignores output other than as defined in the figure when Gate E is not active.

10 RAM

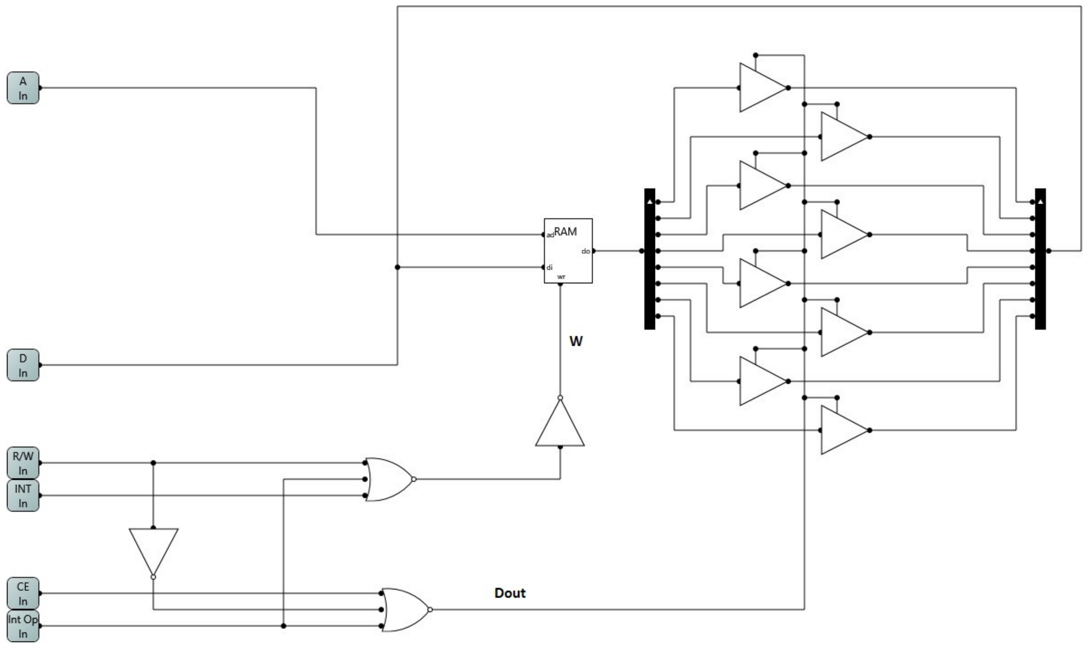


Figure 22 RAM circuit

The RAM (256 x 8 bits) circuit Figure 22 consists of the core memory device, an eight-bit tri-state buffer and some control logic. Practical RAM devices contain the tri-state buffer directly but the LogicCircuit tool requires an external implementation. The address bus is on A and data bus on D with RAM input and output data separated.

RAM function is controlled by the input lines CE, INT, R/W and Int Op as shown in Table 4.

CE	R/W	Int Op	W	Dout	
1	X	X	1	0	RAM not enabled. Tri-state hi-Z
X	X	1	1	0	RAM not enabled. Tri-state hi-Z
0	1	0	1	1	RAM enabled for read. Tri-state enabled.
0	0	0	0	0	RAM enabled for write. Data is clocked into RAM by INT. Tri-state hi-Z

Table 4 RAM function control