Microprocessor Design

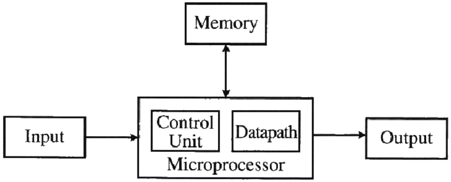
School of Computer Science and Electronic Engineering
University of Essex (UK)

CE339/CE869 - Lecture 7
Jan 2020

Outline

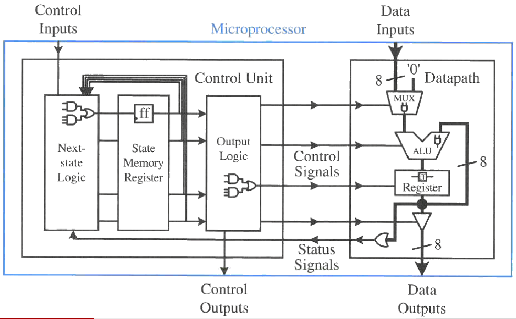
Overview of A Microprocessor

- The Von Neumann model of a computer consists of four main components: the input, the output, the memory, and the microprocessor.
- We are concentrating on the design of digital circuitry of the microprocessor, the memory, and other supporting digital logic circuits.
- The logic circuit for a microprocessor can be divided into two parts: datapath and control unit.



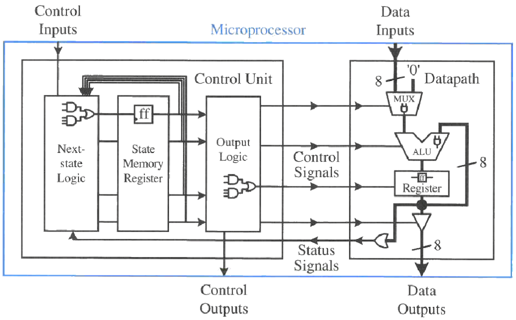
Datapath

- The datapath is responsible for the actual execution of all data operations performed by the microprocessor.
- The ALU and the register are connected together with multiplexers and data signal lines.
- In the following architecture, the output is connected to:
    - the right operand of the ALU.
    - an OR gate used as comparator for the test "not equal to 0".
    - a tri-state buffer.



Control Unit

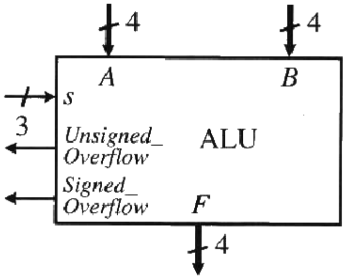
- The control unit controls all of the operations of the datapath and the operation of the entire microprocessor.
- It is a finite state machine (FSM) that executes by going from one state to another and there are only a finite number of state for the machine to go to.
- There are three parts:
  - the next-state logic - to determine what the next state should be.
  - the state memory - to remember the current state that the FSM is in.
  - the output logic - to generate the control signals for controlling the datapath.



Outline

Arithmetic Logic Unit (ALU)

Example of ALU:

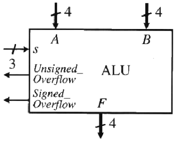


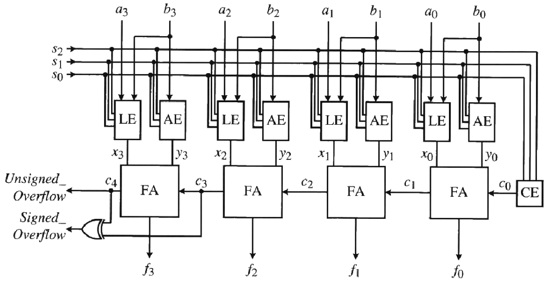
s_2	s_1	s_0	Operation
0	0	0	Pass
0	0	1	AND
0	1	0	OR
0	1	1	NOT
1	0	0	Addition
1	0	1	Subtraction
1	1	0	Increment
1	1	1	Decrement

Arithmetic Logic Unit (ALU)

Example of implementation:

- A four bit adder.
- The logic extender (LE) is for manipulating all logic operations.
- The arithmetic extender (AE) is for manipulating all arithmetic operations.
- The carry extender (CE) is for modifying the primary carry-in signal, so that arithmetic operations are performed correctly.





a_3 b_3 a_2 b_2 a_1 b_1 a_0 b_0
s_2
s_1
s_0
LE
AE
x_3 y_3 x_2 y_2 x_1 y_1 x_0 y_0
Unsigned_Overflow
Signed_Overflow
c_4 c_3 c_2 c_1 c_0
FA
CE
f_3 f_2 f_1 f_0

ALU Truth Table

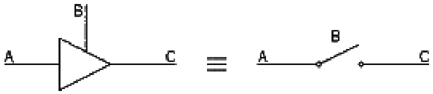
Example of ALU truth table:

s_2	s_1	s_0	Operation	x_i (LE)	$y_i(AE)$	c_0(CE)
0	0	0	Pass	a_i	0	0
0	0	1	AND	a_i AND b_i	0	0
0	1	0	OR	a_i OR b_i	0	0
0	1	1	NOT	a_i'	0	0
1	0	0	Addition	a_i	b_i	0
1	0	1	Subtraction	a_i	b_i'	1
1	1	0	Increment	a_i	0	1
1	1	1	Decrement	a_i	1	0

Tri-State Buffer

- It has three states 0, 1, and Z.
- The value Z represents a high-impedance state, which acts like a switch that is opened.
- It is used to connect several devices to the same bus.
- If two or more devices are connected directly to a bus without using tri-state buffer, signals will get corrupted on the bus.

B	C
0	Z
1	A



Tri-State VHDL

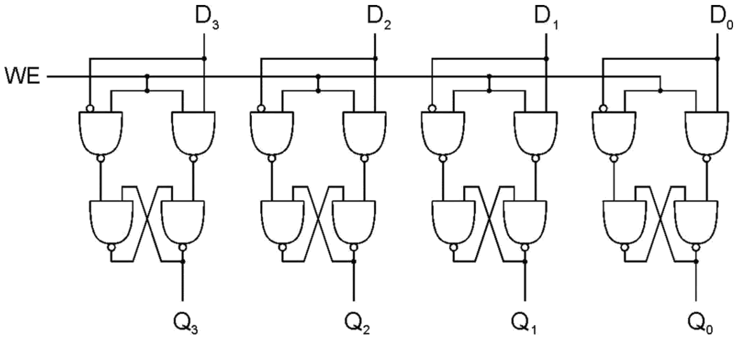
```vhdl
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;

ENTITY TriStateBuffer IS PORT(
   E: IN STD_LOGIC;
   D: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   Y: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END TriStateBuffer;

ARCHITECTURE Behavioral OF TriStateBuffer IS
BEGIN
   Y <= D WHEN E = '1'
        ELSE (OTHERS=>'Z');
END Behavioral;
```

Register

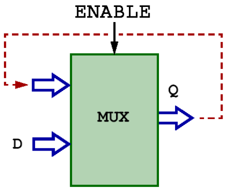
- A register stores a multi-bit value, with enable (load), set, and clear inputs.
- A collection of D-latches, all controlled by a common WE.
- When WE=1, *n*-bit value D is written to register



Latch Inference

- Consider the incomplete **if** statement shown below (i.e. no else part!)
- What happens if the condition is tested and found to be false?
- In this case the value of Q should stay the same.
- Therefore a latch is required to store the value of Q.

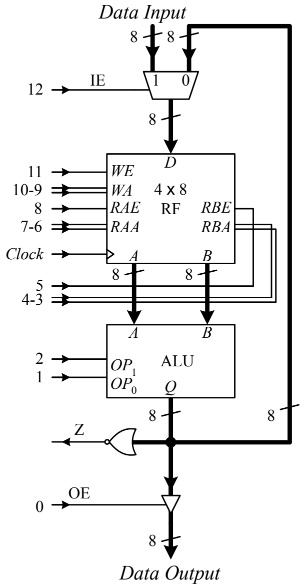
```
if ENABLE = '1' then
    Q <= D;
end if;
```



- If the process is gated by a clock, a set of flip-flops (instead of latches) is inferred.

Register Files

- Register file: an array of registers in a CPU.
- Unlike SRAMs, it has dedicated read and write ports (ordinary SRAMs usually read and write through the same ports).
- It is usually used for the source operands and destination of the ALU and therefore it has one write port and two read ports.
- Since the ALU normally takes two input operands, the register file should be able to output two values from two different locations at the same time.
- Write is synchronous while read is often asynchronous: in a single clock, data can be read, transformed and written back.



Dedicated Register File

- One write port and two read port register file.
- The read ports A and B are connected to the two input operands of the ALU.

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;
USE IEEE.NUMERIC_STD.all;

ENTITY regfile IS PORT (
   clk: IN STD_LOGIC;
   WE:  IN STD_LOGIC;                   -- write enable
   WA:  IN STD_LOGIC_VECTOR(2 DOWNTO 0); -- write address
   RAE: IN STD_LOGIC;                   -- read port A enable
   RAA: IN STD_LOGIC_VECTOR(2 DOWNTO 0); -- read port A address
   RBE: IN STD_LOGIC;                   -- read port B enable
   RBA: IN STD_LOGIC_VECTOR(2 DOWNTO 0); -- read port B address
   input:IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   Aout, Bout:OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END regfile;
```

Dedicated Register File

```
ARCHITECTURE Behavioral of regfile IS
   SUBTYPE reg IS STD_LOGIC_VECTOR(7 DOWNTO 0);
   TYPE regArray IS ARRAY(0 TO 7) OF reg;
   SIGNAL RF: regArray;
BEGIN
   WritePort: PROCESS
   BEGIN
      WAIT UNTIL RISING_EDGE(clk);
      IF WE = '1' THEN
         RF(TO_INTEGER(UNSIGNED(WA))) <= input;
      END IF;
   END PROCESS;

   --ReadPortA
   Aout <= RF(TO_INTEGER(UNSIGNED(RAA))) WHEN RAE = '1'
      ELSE (OTHERS => '0');

   --ReadPortB
   Bout <= RF(TO_INTEGER(UNSIGNED(RBA))) WHEN RBE = '1'
      ELSE (OTHERS => '0');
END Behavioral;
```

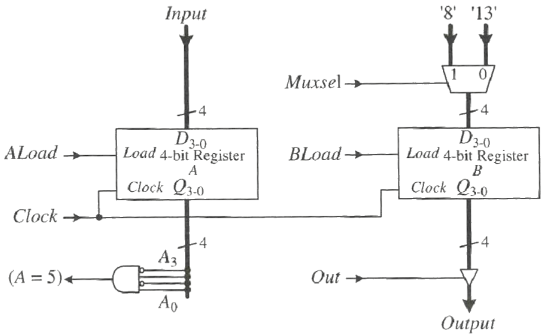
Datapath Design

- A datapath consists of:
  - Functional units (adder, shifter, ALU, comparator, multiplexer, counter).
  - Registers.
  - Buses and tri-state buffer.
- The operations are determined by the control signals from the control unit.
- It must generate status signals from the comparators to the control unit (for jump instructions).
- Here we are concentrating on dedicated datapath. General datapath will be introduced in next lecture.

Dedicated Datapath: IF-THEN-ELSE

- Construct a 4-bit-wide dedicated datapath for solving the IF-THEN-ELSE algorithm

```
INPUT A
IF (A = 5) THEN
    B = 8
ELSE
    B = 13
END IF
OUTPUT B
```



Dedicated Datapath: Control Word

- Control word has one bit for each control signal.
- Each control word takes one clock cycle to perform.
- By combining multiple control words together, the datapath performs the operations in the order given.

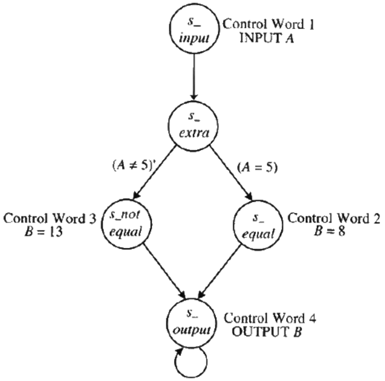
Control word	Instruction	Aload	Muxsel	Bload	Out
1	INPUT A	1	$\times$	0	0
2	B = 8	0	1	1	0
3	B =13	0	0	1	0
4	OUTPUT B	0	$\times$	0	1

Outline

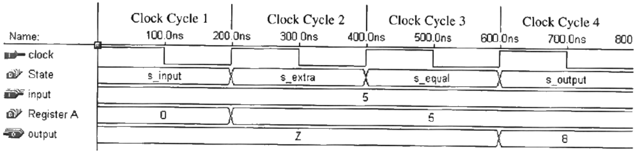
Control Unit

- The **control unit** inside the microprocessor is a FSM.
- It includes the state memory, the next-state logic, and the output logic.
- By stepping through a sequence of states, it controls the operations of the datapath.
- It generates the control signals and controls the sequence of the instructions.
- A state is created for each control word and each state is executed in one clock cycle.

IF-THEN-ELSE: State diagram and Next State Table



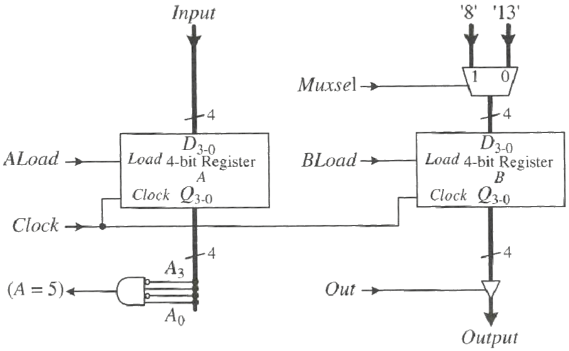
cur_state	next_state	
	(A=5)'	(A=5)
s_input	s_extra	s_extra
s_extra	s_notequal	s_equal
s_notequal	s_output	s_output
s_equal	s_output	s_output
s_output	s_output	s_output



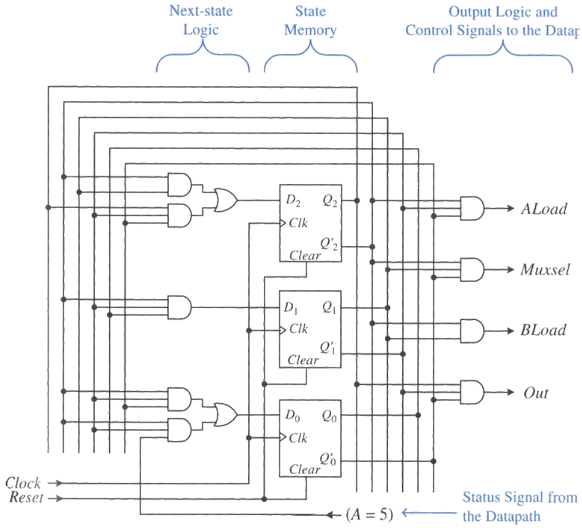
IF-THEN-ELSE: Output, Control Words

Outputs
(control words):

cur_state	CW
s_input	1X00
s_extra	0X00
s_equal	0110
s_notequal	0010
s_output	0X01



IF-THEN-ELSE: Control Unit's Schematics



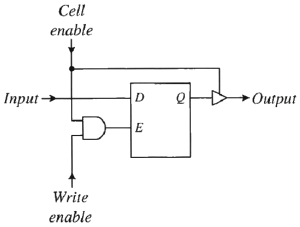
Outline

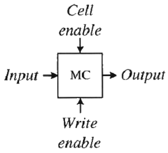
Memory

- Two basic kinds of RAM (Random Access Memory)
  - Static RAM (SRAM)
    - Memory cell uses flip-flop to store bit;
    - Holds data as long as power supplied;
    - Fast.
  - Dynamic RAM (DRAM)
    - Memory cell uses MOS transistor and capacitor to store bit;
    - More compact than SRAM;
    - Slower but denser;
    - Must be periodically refreshed due to capacitor leak - word's cells refreshed when read;
    - Typical refresh rate 15.625μs.
- Non-volatile memories: ROM, EPROM, flash, ...

RAM: "Random-Access" Memory

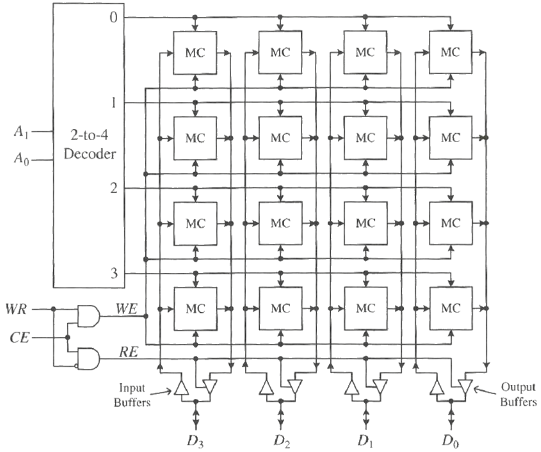
- A common data bus for both reading from and writing to data to the memory.
- Just one data port.
- The number of address lines (n) is dependent on how many locations are in the memory chip, usually a power of 2.
- Chip enable (CE)
- One bit memory cell





Static RAM

- 4×4 static RAM chip.
- Each row forms a single storage location.
- The number of memory cells in a row determines the bit width of each location.
- A 2-to-4 decoder is used to decode the four address location.



RAM VHDL Entity

- 16×4 RAM with 4 address lines and 4 data lines.
- The bi-directional data port is declared as INOUT - can be read from and written to.
- The actual memory content is stored in the variable mem, which is an array of type STD_LOGIC_VECTOR.

```vhdl
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;
USE IEEE.NUMERIC_STD.all;

ENTITY memory IS PORT(
   CE, WR: IN STD_LOGIC;     -- chip enable, write enable
   A: IN STD_LOGIC_VECTOR(3 DOWNTO 0); -- address
   D: INOUT STD_LOGIC_VECTOR(3 DOWNTO 0)); -- data
END memory;
```

RAM VHDL Architecture

```
ARCHITECTURE Behavioral OF memory IS
BEGIN
   PROCESS(CE,WR)
      SUBTYPE cell IS STD_LOGIC_VECTOR(3 DOWNTO 0);
      TYPE memArray IS ARRAY(0 TO 15) OF cell;
      VARIABLE mem:  memArray;
   BEGIN
      IF CE = '1' AND WR = '0' THEN     -- read
         D <= mem(TO_INTEGER(UNSIGNED(A)));
      ELSIF CE = '1' AND WR = '1' THEN -- write
         mem(TO_INTEGER(UNSIGNED(A))) := D;
      ELSE -- not enabled or invalid
         D <= (OTHERS=>'Z');
      END IF;
   END PROCESS;
END Behavioral;
```

RAM VHDL Architecture

```vhdl
ARCHITECTURE Behavioral OF memory IS
BEGIN
   PROCESS(CE,WR)
      SUBTYPE cell IS STD_LOGIC_VECTOR(3 DOWNTO 0);
      TYPE memArray IS ARRAY(0 TO 15) OF cell;
      VARIABLE mem:  memArray;
   BEGIN
      IF CE = '1' AND WR = '0' THEN    -- read
         D <= mem(TO_INTEGER(UNSIGNED(A)));
      ELSIF CE = '1' AND WR = '1' THEN -- write
         mem(TO_INTEGER(UNSIGNED(A))) := D;
      ELSE -- not enabled or invalid
         D <= (OTHERS=>'Z');
      END IF;
   END PROCESS;
END Behavioral;
```

Representing a RAM memory by an array of bytes is a perfectly reasonable behavioural model, but it won't necessarily synthesise into a memory (it might synthesise into a large block of registers or latches!).

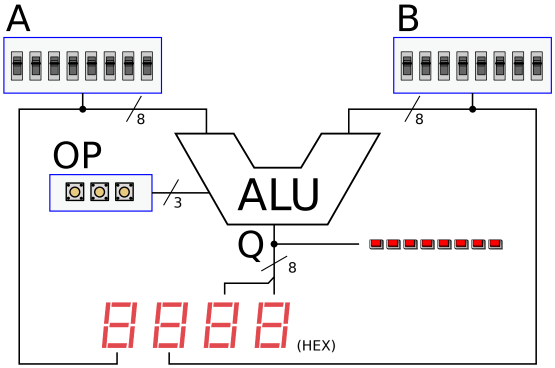
ROM: “Read-Only” Memory

- Nonvolatile memory
- Can be read from but not written to, by a processor in an embedded system.
- Traditionally written to, “programmed”, before inserting to embedded system.
- Usage
    - store software program for general-purpose processor;
    - program instructions can be one or more ROM words;
    - store constant data needed by system;
    - implement combinational circuit.

Outline

Lab ALU

Implement (using behavioural style) an ALU performing the operations in the table on the left and test it on the Basys 3 board using the layout on the right:



OP	Operation
000	Addition
001	Subtraction
010	Increment A
011	Decrement A
100	LT (nonzero if A $<$ B)
101	NOT A
110	Logic AND
111	Logic OR

Thank you
for your attention

Questions?

Microprocessor Design

School of Computer Science and Electronic Engineering
University of Essex (UK)

CE339/CE869 - Lecture 7
Jan 2020