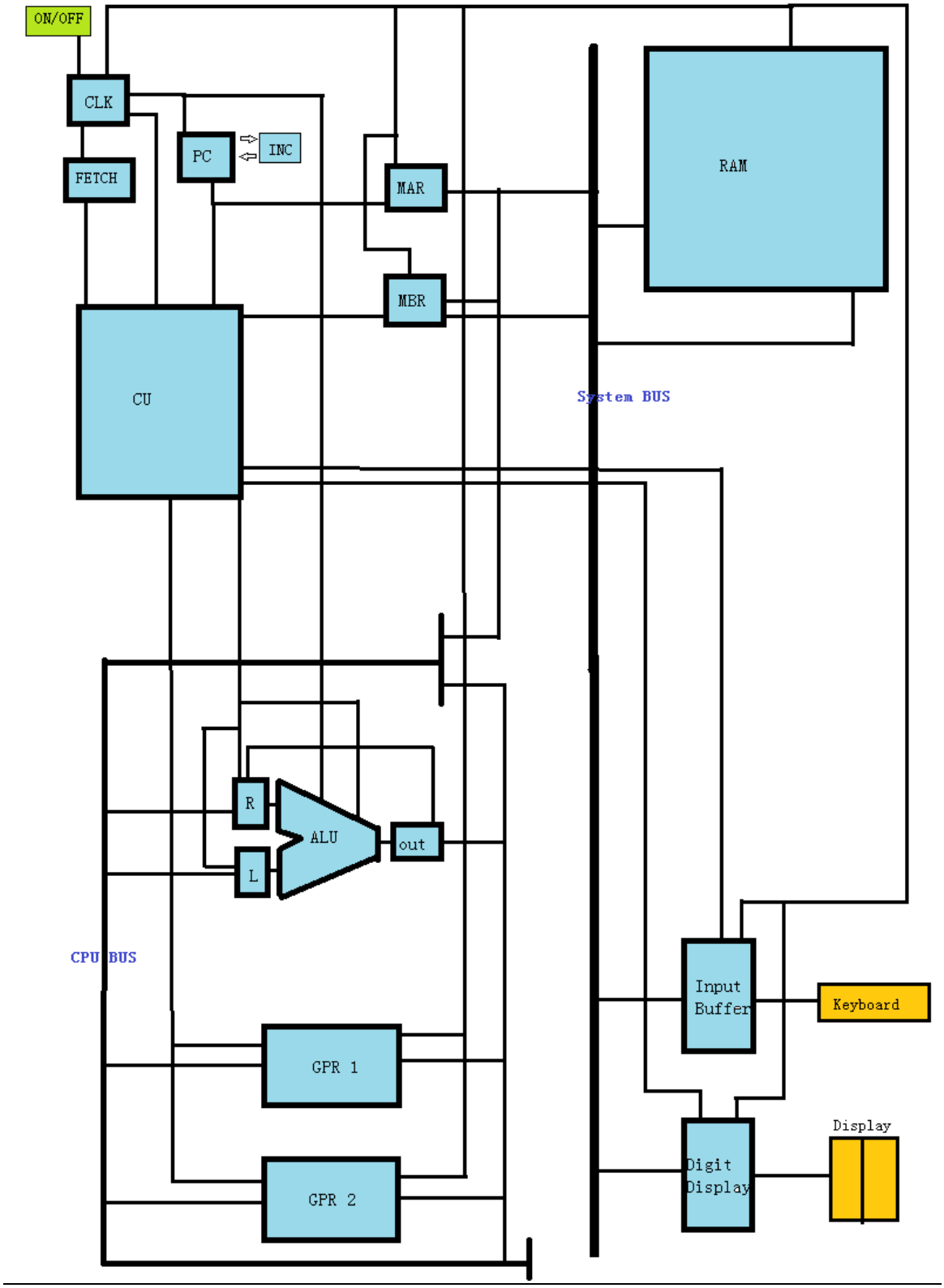
Classical CPU Project

Super Kawaii Marshmallow Duo

By:
Xu Yi Tian (260520039)
Yang Ye (260426571)

COMP 273 – Introduction to Computer System
Instructor: Prof. Joseph Vybihal
April 16th, 2013
McGill University

Section 1.1 CPU Diagram



Section 1.2 Functionality and Flow

The CPU has four main steps: FETCH, DECODE, EXECUTE, STORE RESULT.

The FETCH step starts once the computer is turned on, and registers in CPU are all initially set to zero. Program counter (PC) starts with first address in RAM (0000), sending it to memory address register (RAM). After 4-bit ALU increments the PC by 1, MAR passes the address to the System BUS, which spreads it across the system board. Gates in RAM will open and its address register (AR) will receive the address. Through an internal built-in routine, RAM will pass the content stored at the specific address to its data register (DR), which spits it out onto the System BUS. The FETCH stage ends with the byte passing from the memory buffer register and entering the instruction register (IR).

The DECODE stage starts on IR by splitting each bit of the input byte into two, one indicating that the bite is a “1”, the other, that the bit is a “0”. These bits will be evaluated by the control unit (CU). Each of the 11 instructions is represented by a unique combination of those bits, detected through AND gates. When an op-code shows up, the corresponding AND gate opens and then send signal to later steps in the CU for respective instruction routines.

To execute, the instruction is controlled by 13 1-bit binary outputs signs. Most of the instruction will use more than one output in order to complete the task. These outputs are released in three main ways: READ/WRITE, RESET, and INPUT/OUTPUT. For instance, READ/WRITE is usually used for loading and storing into registers and RAM. When it is turned on, it set the targeted register to write mode, otherwise leave it in read mode. RESET is used for resenting ALU and MBR. The INPUT/OUTPUT sends instructions to the display and input buffer.

Instructions that require more than one output to complete the task need some way of synchronization. To allow delay outputs from the control unit, required information is repeatedly stored in different AND gates and registers and then released at the next clock tick.

Finally, when instruction STR appears, the result temporarily stored in a general purpose register is sent to RAM through MBR and MAR.

At the end of each instruction, CU will send signal to restart FETCH.

Section 2.1 Detailed descriptions

ON button

The user must press ON button to turn on the computer, otherwise the system will not respond to any instruction. When it is pressed, a D Flip-Flop will store a 1, which is outputted to an AND gate that enables clock ticks. At the same time, it sets PC and all the other registers to 0 and turn on some of the registers and gate that will be needed during the entire execution of the program. Clicking the button again would reset all register to zero.

System BUS:

The System BUS has 13 bits; first bit, control bus, is for the R/W mode, and this is used to connect the input R/W of RAM and output of IR. The following 8 bits is the data bus, which is used for the input and output data, and they are connected by DR, two digit hexadecimal display input, 8-bit input buffer, and MBR. The last four bits is the address bus which is connected by MAR and AR.

CPU BUS:

The CPU BUS has 8 bits; they carries input and output data, and are connected by MBR, all the registers for ALU (Left, Right and ALU-Out registers), and the two general purpose registers. To make sure that MBR passes data to one of the appropriate BUS, a series of demultiplexers controlled by the CU lies between the two BUSes to enable selection of the data path.

Clock:

The main clock starts ticking when the ON switch is pressed. It is connected to all the black-boxes except general purpose register, MAR and MBR. A series of D Flip-Flops connected to the clock to organize the flow for the FETCH stage. The instruction cycle is represented by six outputs on the right connect to the basic operation of the system. Another series of D Flip-Flops below take care of the BZ (branch equal) instruction when the PC jumps to the indicated address. Other series of D Flip-Flops for other routines are built inside the respective black-box, also connected to the main clock. There are three inputs from the CU: HALT signal to turn off the clock and reset all registers, BZ signal and restart signal for the FETCH cycle.

CU (Control Unit):

The control unit has an 8-bit instruction register (IR), and contains instructions to execute 11 OP-CODE which are described by mnemonics as: STOP (stop program), COMP (2's compliment), INP (input from keyboard), PTR (print to digital display), SUB (subtraction), ADD (addition), BZ (branch equal), STR (store in RAM), LD (load in RAM) and HALT (shut down computer). The CU decodes the instruction, and sends the signal to other units for execution.

PC (Program Counter):

The PC is a 4-bit register. The instruction pointer is incremented after fetching the instruction, and points to the address of the next instruction that shall be executed. Our PC adder is a series of 4 full-adders such that the first full-adder is connected to an constant 1, thus allowing the PC to be incrementing by 1 each time.

MAR (Memory Address Register):

MAR is a 4-bit register. It is used for either store the memory address from which instruction is fetched or the address to which data for the execution of an instruction will be loaded or stored. The inputs are connected from PC or CU and output, onto the System BUS.

MBR (Memory Buffer Register):

MBR is an 8-bit register. It acts as a buffer and stores the data being transferred to and from the immediate access store. Both the input and output are connected to the CPU BUS and the System BUS.

General Registers:

There are two general 8-bit registers R0 and R1. They are the same black-box used for all the 8-bit registers, each of them containing four inputs for R/W, address, reset and clock ticks. The inputs and outputs of general registers are each connected to the CPU bus.

ALU:

The ALU can perform addiction (ADD) and 2's compliment (COMP). It contains two 8 bit input registers (Left and Right), an A-OUT 8-bit output register, 8 full-adders, 8-half adders, status, negative and zero signal output. Instructions of operation involving one register (such as COMP) will only use the Left register. To execute COMP, an input from the CU will flip all the bits outputted from the Left register and turn on one of the AND gates from which a constant 1 is connected to. The first full-adder thus adds a 1 to complete 2's compliment. To deal with instructions that need both the 2's compliment of both register, two signals from CU are needed to open both AND gate so that 1 can be added twice in the final result. However, such instruction does not exist yet in our CPU.

By the implementation of 2's compliment, subtraction (SUB) and branch equal (BZ) can also be done using this ALU. When the result is zero, a signal is sent to CU to indicate that the PC must jump.

RAM:

The RAM has full R/W functionality, composed of 16 8-bit registers, an 8-bit data register, a 4-bit address register and a R/W mode register. It has input for reset, an signal from clock or CU to indication execution of the loading/fetching or save processes depending on the R/W mode.

Two-Digit Digital Display:

The digital display is represented by two digits of hexadecimal numbers. It contains an 8-bit register that will load the data from the System BUS when it receives signal from CU. The instruction in assembly for the digital display is PRT.

8-Bit Input Buffer:

10 input pins from 0 to 9 are connected to the input buffers which will constantly update the content of its 4-bit register as the clock ticks. It stores the number that the use inputs from the pins and passes it to the first four data wires of the System BUS when it receives signal from the CU. The instruction in assembly for the input buffer is INP.

Section 3.1 Instructions in assembly

ADD R1, R0
-add the contents in general register R1 and R0, and then store the result into R1

STR R1, Address
-store the content of R1 into the given address in the RAM

LD R1, Address
-load the value into R1from the given address in the RAM

BZ R1, R0, Address
-branch to given address if the content in R1 is equal to the content in R0

PRT R1
-output the content in R1 with Digital Display

INP R1
-store the input into R1 using the input buffer

STOP
-mysteriously stops the program whenever this instruction is sent

SUB R1, R0
-subtract the contents in general register R0 from R1, and then store the result into R1

COMP R1
-implement the value of R1 and save back to R1

HALT
-turns off the computer

EXTRA BUTTONS (on RAM)
-Notice that we have set some extra buttons around RAM. There are two buttons on the left side of RAM. The one of them has “program” labeled. When it is pressed, it sets the two-number-multiplication program into RAM. Above, another button has “R memory” labeled, which serve to reset memory in RAM. This function is necessary to re-run the program because some constant that is set along with the program (on pressing “program”) may have been changed during the execution.

-On the other side, four buttons labeled from 1 to 128 are connected to the last memory space in RAM. This is convenient for setting the constant when running the program. One can press for any combination to give the desired number rather than set the input by accessing into the RAM memory.

Section 3.2 Instruction bit formatting

General Register: represented by 1 binary
R1:1
R2:0

Address: represented by four binary bits, and we can store maximum of 16 addresses

ADD: 001000 X X
follow by two registers

STR: 011 XXXX X
follow by one 4 bits address and one 1 bit register

LD: 010 XXXX X
follow by one 4 bits address and one 1 bit register

BZ: 10 XXXX X X
follow by one 4 bit address and two 1 bit registers

PRT: 0001001 X
follow by one 1 bit register

INP: 0001000 X
follow by one 1 bit register

STOP: 0000 XXXX

SUB: 001001 X X
follow by two 1 bit registers

COMP: 0001001 X
follow by one 1 bit register

HALT: 11 XXXXXX

Section 3.3 Description of execution of instructions over the CPU

HALT: is used for turn off the program. It is completed by negating the ON/OFF register in the clock, and disabling clock ticks and closing gates.

STOP: is intended to mysteriously stop the program. This is done by disabling the FETCH step, through the CU sending no signal to the clock.

ADD, SUB: does arithmetic operation with the content store in the two register indicated by the last two bits of the instruction, and execute in following steps:
-send "2's complement" signal for the Left register of ALU if the instruction is SUB
-load the content of the first register to the Right register of the ALU; ALU send signal back to CU when it is done
-load the content of the second register to the Left register of the ALU; again ALU send signal back when finished
-send signal to ALU to execute the operation, result stored in A-OUT
-save the result into the first register
-reset the ALU

COMP: turns the contentment of the register indicated by the last bit in the instruction into 2's compliment form by the following steps:
-send "2's complement" signal for the Left register of ALU
-load the content of the first register to the Left register of the ALU; ALU send signal back to CU when it is done
-send signal to ALU to execute operation
-save the result into the first register
-reset the ALU

LOAD: reads the address from the 4^{th} to the 7^{th} bit of the given instruction, and then save the content in RAM into the register indicated on the last bit of the instruction by the following steps:
-send address to the System BUS through MAR and open gate to System BUS
-send signal to RAM, which sends data back on the System BUS
-store data in MBR and open gate to CPU BUS
-store data in register

STR: loads the content of register indicated on the last bit of the instruction and write the content to RAM in the given address (bit 4-7) by the steps below:
-send Write signal to RAM
-send register content and address to the System BUS through MBR and MAR and open gate to System BUS
-send signal to RAM to execute operation

BZ: does subtraction with the content of the register indicated by the 2 last bits of the instruction and pass the address (bit 3-6) to the PC if the result is zero through the following steps:
-does all the steps in SUB except the saving part
-if ALU send the “zero” signal back to CU, CU send signal to the clock so the address indicated in the instruction can be send and store in PC
-otherwise, CU does not do anything and the program continues

PRT: displays the content of the register indicated by the last bit of the instruction by the following steps:
-send register content to the System BUS through MBR and open gate to System BUS
-send signal to digital display to display content on the System BUS

INP: gets the input data from keyboard and save to the register indicated by the last bit of the instruction through the following steps:
-open gate to System BUS and send signal to input buffer
-store data to MBR and open gates to CPU BUS
-store data to register

Section 3.4 Program & Instructions

Address	Label	Assembly language	Binary instruction
0000	main	INP R2	0001 0000
0001		BZ R1 R2 END	1011 0010
0010		LD R1 result	0101 1101
0011		LD R2 input1	0101 1110
0100		ADD R1 R2	0010 0010
0101		STR R1 result	0111 1101
0110		PRT R1	0001 0011
0111		LD R1 counter	0101 1011
1000		LD R2 constant	0101 1000
1001		ADD R1 R2	0010 0010
1010		STR R1 counter	0111 1011
1011		BZ R1 R1 main	1000 0011
1100	END/constant	STOP/1	0000 0001
1101	counter	Start on 0	0000 0000
1110	result	Start on 0	0000 0000
1111	input 1		

At address 1100, the data stored in RAM act both as a constant and an instruction.

Section 4: Appendix – Procedures to Run the Program

Once the circuit is opened on LOGISIM, press on the ON button on the top left corner to start up the computer (do not enable ticks). To run the program, click on the button labeled “program” next to RAM in the upper right corner to load the program into RAM. By default, the number in RAM used to perform the calculation is 0. Click on the buttons on the right side of RAM to set a non-zero value (each button is connected to one of the bits in the last byte in RAM).

To input a number from keyboard, turn on only one of the input pins connected to the buffer below RAM. Keep it on during the entire execution of the program.

Once the above the all set, enable ticks to run. The program finishes when only the clock is ticking. At this point, the result should be displayed on the digital display below the input buffer.

Reset RAM, the input pin (if the input number has changed) and the PC (from the ON button) to re-run the program. More precisely, click on “R memory” before loading the program in RAM as explained previously and press on the ON button (which also resets all registers) before re-running the program.