Al-alamein international university

Computer architecture course

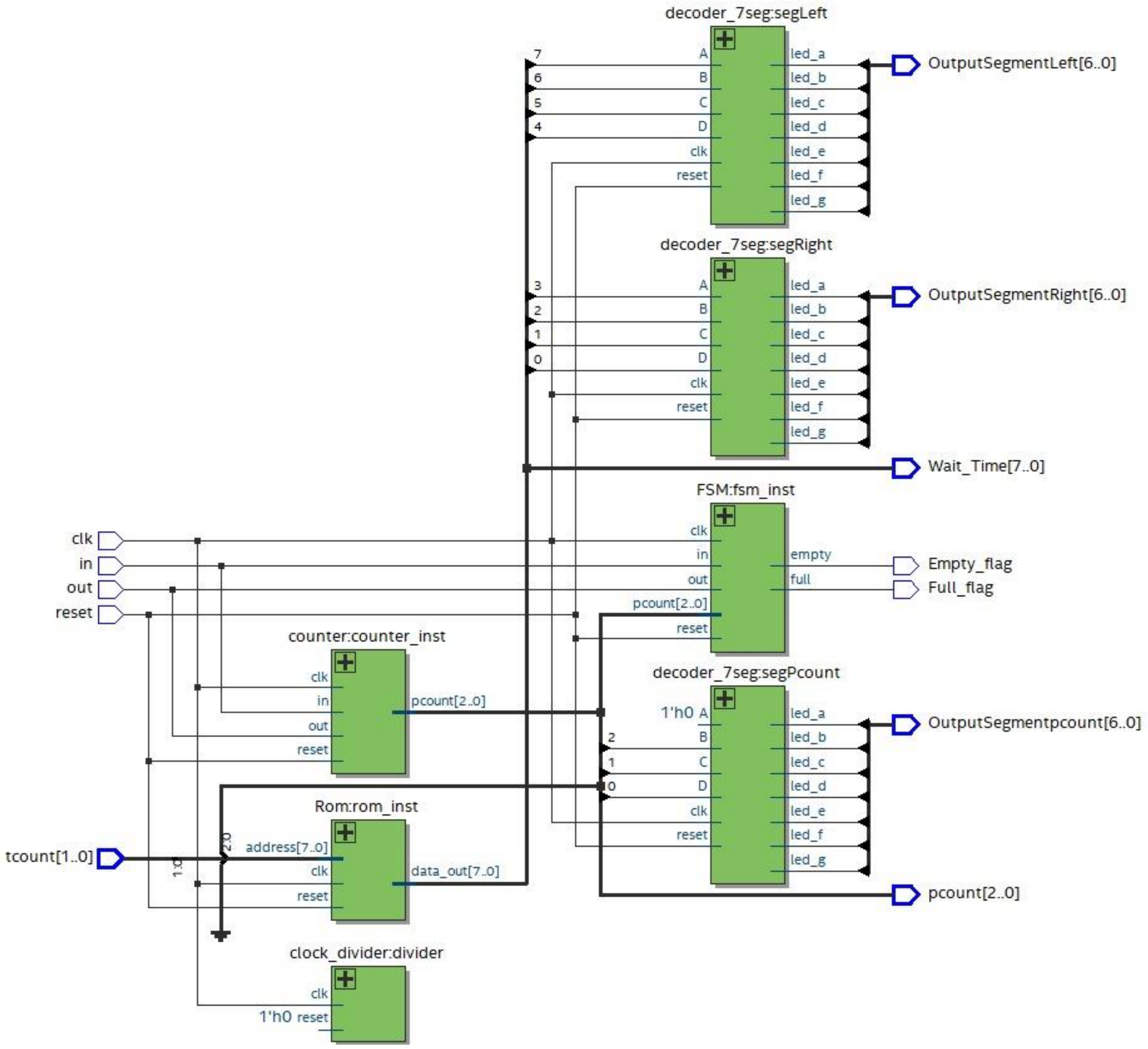
ABQM Project

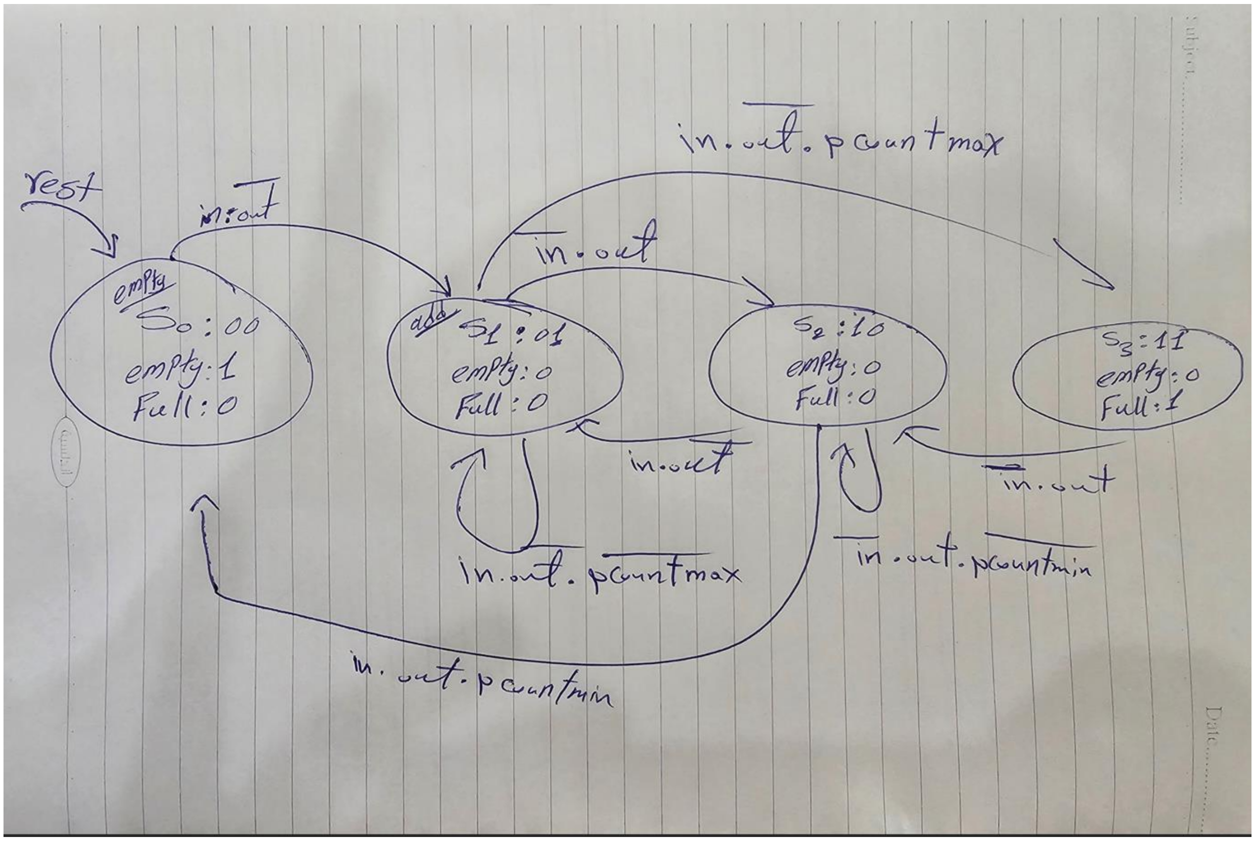
The first person in line is the first person served
The last person in line is the last person served

Project code explanation:

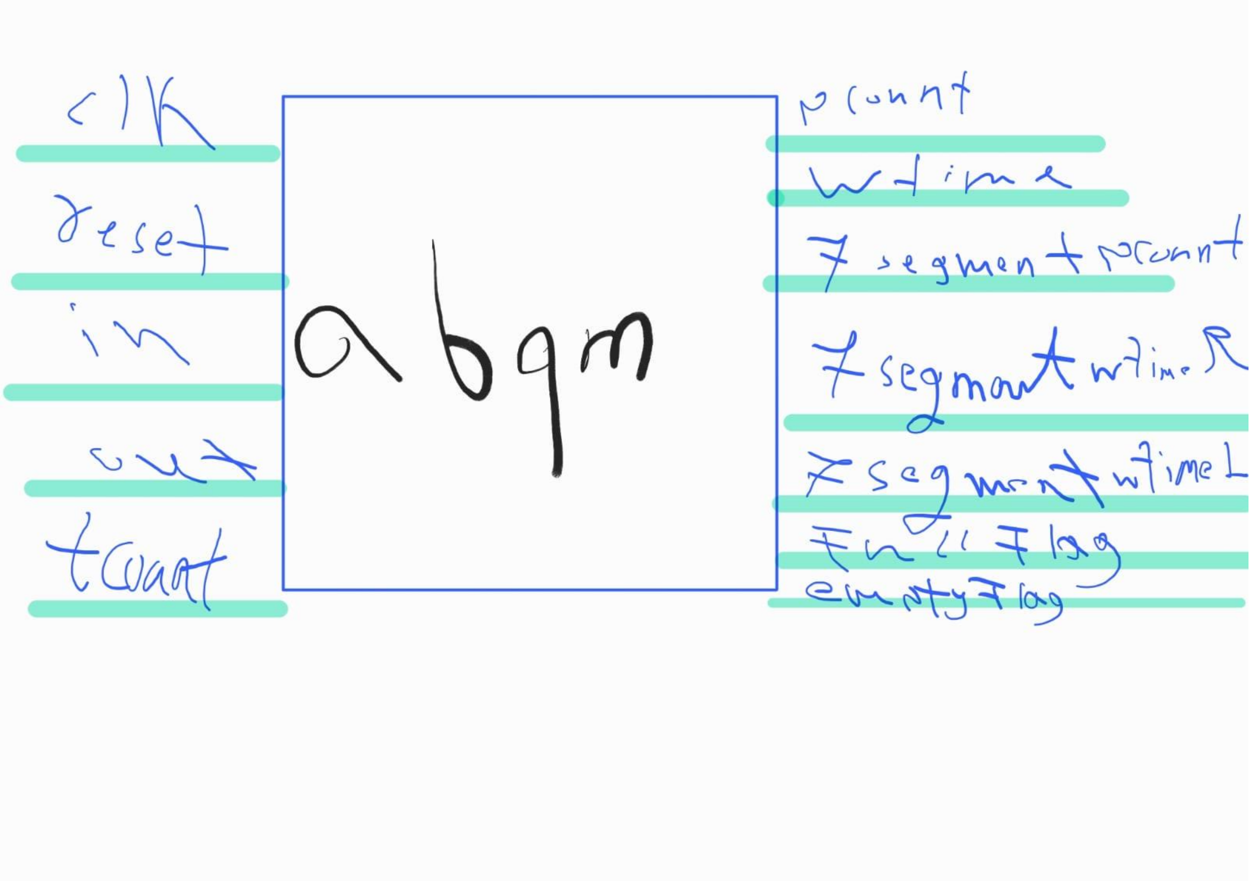
The project consists of 6 modules:

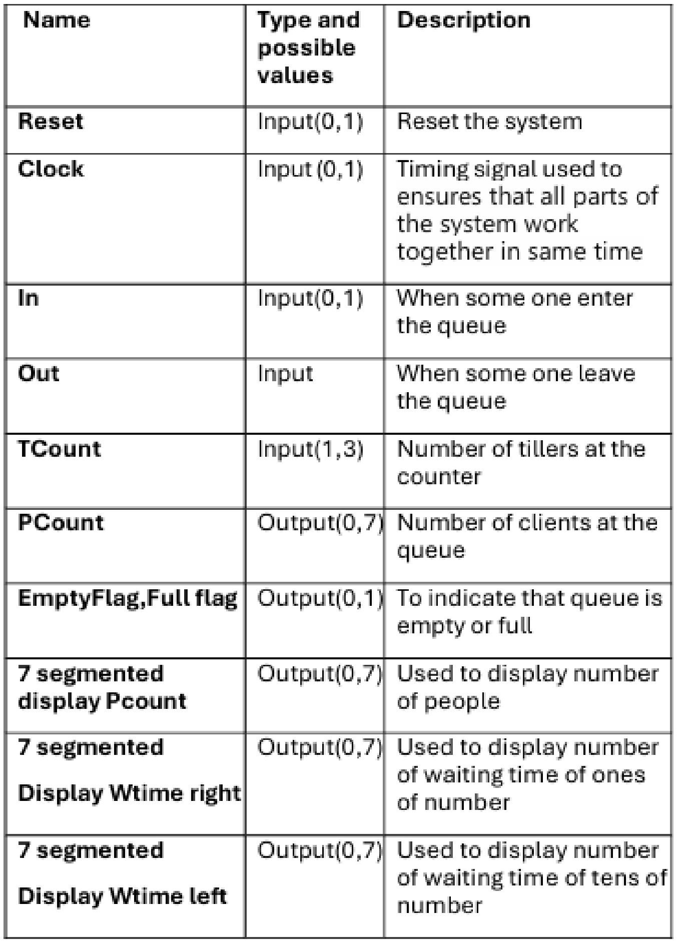
The main module (abqm) . . (Digital clock) module . . (FSM) module . . (clock divider) module . . (counter) module . . (Rom)




rest
in.out
empty
S0 : 00
empty:1
Full:0
in.out.pcountmax
in.out
add
S1 : 01
empty:0
Full:0
S2 : 10
empty:0
Full:0
S3 : 11
empty:0
Full:1
in.out
in.out
in.out.pcountmax
in.out.pcountmin
in.out.pcountmin

Present State	Next State ……………………..	In ………………….	Out ………………	pcount (value)	Full ……………….	Empty ……………..
s0 (Empty)	s1 (Add)	1	0	X	0	0
s0 (Empty)	__	0	0	X	0	1
s1 (Add)	s1 (Add)	1	0	! 3b’111	0	0
s1 (Add)	s2 (Subtract)	0	1	X	0	0
s2 (Subtract)	s0 (Empty)	0	1	3b’000	0	0
s2 (Subtract)	s2 (Subtract)	0	1	! 3b’000	0	0
s2 (Subtract)	s1 (Add)	1	0	X	0	0
s3 (Full)	s2 (Subtract)	0	1	X	0	0
s1(add)	s3(full)	1	0	3b’111	1	0


clk
reset
in
out
tcount
abqm
pcount
wtime
7segment pcount
7segment wtime R
7segment wtime L
Full Flag
emptyFlag



Name	Type and possible values	Description
Reset	Input(0,1)	Reset the system
Clock	Input (0,1)	Timing signal used to ensures that all parts of the system work together in same time
In	Input(0,1)	When some one enter the queue
Out	Input	When some one leave the queue
TCount	Input(1,3)	Number of tillers at the counter
PCount	Output(0,7)	Number of clients at the queue
EmptyFlag,Full flag	Output(0,1)	To indicate that queue is empty or full
7 segmented display Pcount	Output(0,7)	Used to display number of people
7 segmented **Display Wtime right**	Output(0,7)	Used to display number of waiting time of ones of number
7 segmented **Display Wtime left**	Output(0,7)	Used to display number of waiting time of tens of number

Detailed Specifications of ABQM™:

1. Photocells: Both ends of the queue have photocells. Each photocell outputs a logic 1 if the light beam is uninterrupted and switches to logic 0 when the beam is interrupted.

2. Queue Management: Clients enter the queue from the back and leave from the front.

3. Display Information: The number of people in the queue (Pcount) and the expected waiting time (Wtime) are displayed on a Seven Segment Display and empty,full flag.

4. Pcount Adjustment: Pcount increases by one when a client enters the queue and decreases by one when a client leaves, regardless of how long the client stands in front of the photocell.

5. Waiting Time Calculation: Wtime is calculated as follows:

• If Pcount ≠ 0, Wtime = 3*(Pcount + Tcount – 1) / Tcount, where Tcount (1, 2, or 3) is the number of tellers in service. The fractional part of Wtime is ignored.

6.Status Flags: ABQM™ has binary flags indicating whether the queue is empty or full.

7.System Reset: The system can be reset, by making the full flag is equal to zero . . and Pcount while setting the empty flag to 1.

FSM module:

1. Module Declaration:
   - The module has input ports clk, reset, in, and out, and output ports pcount, full, and empty.
2. State Declaration:
   - A register state is declared to store the current state of the FSM.
   - The state register is 2 bits wide, indicating that there are four possible states: s0, s1, s2, and s3.
3. Output Registers:
   - Output registers full and empty are declared to indicate whether the FSM is in a full or empty state, respectively.
4. State Definitions:
   - State definitions are declared using parameters. Each state is assigned a two-bit binary value.
   - s0 represents the empty state (00), s1 represents the add state (01), s2 represents the subtract state (10), and s3 represents the full state (11).
5. always Block:
   - The always block is sensitive to the positive edge of the clock (posedge clk) or the positive edge of the reset signal (posedge reset).

- Inside the `always` block, the behavior of the FSM is defined based on the current state and input signals.

6. Reset Condition:
    - If the `reset` signal is asserted (`reset == 1`), the FSM is reset to the initial state (`state = 0`).
    - The `empty` flag is set to `1` to indicate an empty state, and the `full` flag is set to `0`.

abqm module:

1. `abqm` Module:
    - Inputs: `clk`, `reset`, `in`, `out`, and `tcount`.
    - Outputs: `Full_flag`, `Empty_flag`, `OutputSegmentRight`, `OutputSegmentLeft`, `OutputSegmentpcount`, `Wait_Time`, and `pcount`.
    -

Digital clock :

The code you provided consists of two modules:

1. decoder_7seg:

This module is a decoder for a 7-segment LED display. It takes a 4-bit binary input (A, B, C, D) representing a digit and drives the individual segments (led_a to led_g) of the display to show the corresponding number (0-9) or letter (A-F).

- **Inputs:**
  - clk (clock): Clock signal for synchronization.
  - reset: Active high reset signal to initialize the display.
  - A, B, C, D: 4-bit binary input representing the digit or letter to display.
- **Outputs:**
  - led_a to led_g: Individual outputs that control the segments of the 7-segment display.

The logic uses a series of inverters and AND/OR gates to decode the 4-bit input into the appropriate segment patterns for each digit or letter.

Counter:

This module implements a counter with a maximum capacity of 7 (represented by a 3-bit register `pcount`). It takes the following signals:

- **clk (clock):** A clock signal that triggers the counter's operation on its positive edge.
- **reset (active high):** A reset signal that initializes the counter to 0 (3'b000) when asserted.
- **in:** An input signal that controls incrementing the counter.
- **out:** An input signal that controls decrementing the counter.

The counter has a single output:

- **pcount (output reg [2:0]):** A 3-bit register that stores the current count value (0 to 7).

Here's how the code functions:

1. **Always Block:** The entire logic resides within an `always` block that executes on every positive edge of the clock (posedge clk) or when the reset signal is asserted (posedge reset).
2. **Reset:** If the reset signal is active (high), the `pcount` register is set to 0 (3'b000), effectively resetting the counter.
3. **Increment:** If the `in` signal is active (high) and the current count (`pcount`) is less than the maximum value (3'b111), the counter increments by 1. This ensures the counter doesn't overflow beyond its capacity.
4. **Decrement:** If the `out` signal is active (high) and the current count (`pcount`) is greater than the minimum value (3'b000), the counter decrements by 1. This prevents the counter from underflowing below 0.

Rom:

This module called Rom. This module acts as a Read-Only Memory (ROM). Here's a breakdown of its functionality:

Inputs:

- **clk (clock):** A clock signal that triggers the ROM operation on its positive edge.
- **reset (active high):** A reset signal that initializes the ROM's output to 0 (all bits set to 0) when asserted.
- **address (input [7:0]):** An 8-bit address that specifies the location of the data to be read from the ROM.

Output:

- **data_out (output reg [7:0]):** An 8-bit register that stores the data read from the ROM based on the provided address.

Functionality:

1. **Always Block:** The logic resides within an always block that executes on every positive edge of the clock (posedge clk) or when the reset signal is asserted (posedge reset).
2. **Reset:** If the reset signal is active (high), the data_out register is set to 0 (8'b00000000), effectively clearing the output.

Team members:

Youssif Yasser said	22100809
Mohammed ahmed shoukry	22100259
Mahmoud eid Khamis	22100680