**LASTNAME (in UPPERCASE)** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, First name\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**LASTNAME (in UPPERCASE)** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, First name\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Quiz 3 (10 pts)** **Convert the C-like code provided to a High-Level State Machine**. Do not optimize or change the code, simply convert as provided.

**Using the RTL design method, convert the high-level state machine you designed to a Datapath and Controller**. You do not need to provide the Controller’s corresponding Boolean equations (i.e. you only need to provide the FSM). Note: Be sure to explicitly list all inputs, outputs, and local registers along their respective sizes in your design using as few bits as possible for each register.

**Inputs: a[256] (8 bits), go (1-bit)**

**Outputs: max\_diff (1-byte), done (1-bit)**

**MAX\_DIFF:**

**while(1) {**

**while(!go);**

**done = 0;**

**i = 0;**

**max = 0;**

**min = 255; // largest 8-bit value**

**while( i < 256 ) {**

**if( a[i] < min ) {**

**min = a[i];**

**}**

**if( a[i] > max ) {**

**max = a[i];**

**}**

**i = i + 1;**

**}**

**max\_diff = max - min;**

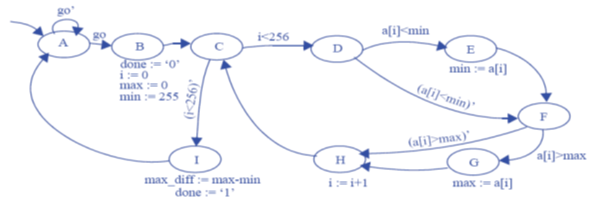
**done = 1;**

**}**

Inputs: go(1 bit), a (256-byte register file)

Outputs: done (1 bit), max\_diff (8 bits)

Local registers: min (8 bits), max( 8 bits), **i (9 bits) – 9 bits are used so that it can count up to 256 = 10000000 in binary = 100 in hexadecimal**





9

8

8

9

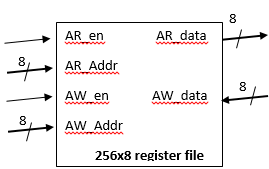
8

8

**a[i]**

**i[7:0]**

**AR\_en**







AR\_en = 1

AR\_en = 1

AR\_en = 1

AR\_en = 0

AR\_en = 1