1. An MPU has a 16-bit unidirectional address and an 8-bit bidirectional data bus multiplexed with the lower 8 bits of its address as shown below:

Where:
- ALE is Address Latch Enable (active high)
- WR is Write Enable (active low)
- RD is Read Enable (active low)
- T \( \overline{R} \) is Transmit (active high) and Receive (active low)

MPU’s bus cycle consists of 4 clock cycles: T1, T2, T3 and T4. Read and write bus cycles are shown below. Note that address, data and control signals are dispatched at the positive edge of clock.
This MPU interfaces with 4 Flash EPROMs. Each EPROM stores valid data at the positive edge of $WR$. $WR$ equals to 0 only when valid data is available from MPU. Write access time to write valid data to EPROM takes 1 clock cycle.

As soon as the valid address is latched, $RD$ transitions to 0. Read access to read valid data from EPROM takes 2 clock cycles. Address hold to output delay is 0 cycles during read. Note that read and write to Flash EPROM takes place with respect to the positive edge of clock.

25pts (a) Show the timing diagram that includes MPU address, data and control signals for a write bus cycle followed by a read bus cycle.

25pts (b) Show the overall schematic of this MPU and 4 Flash EPROMs. Make sure the entire address space is used by the Flash EPROMs. Draw the address latch and transceiver in this system in gate level. Each EPROM has address and data ports, $WE$ (active low write enable), $OE$ (active low output enable) and $CE$ (active low chip enable) control inputs. Indicate all bus widths on your schematic.
Diagram 1:

- MPU
- ALE
- WR
- T/R
- RD
- Clock
- A15-A8
- AD7-ADO

(a)

Clock:
- T1, T2, T3, T4

A15-A8:
- Write
- Read

AD7-AD0:
- Write
- DWrite
- Read
- DRead

ALE:
- 25 MHz

WR:
- RD:

Bus cycle:
- ≤ 2 clocks
2. Data values of FH, EH, DH, CH, BH, AH, 9H, 8H, 7H, 6H, 5H, 4H, 3H, 2H, 1H and 0H are written into DRAM addresses of C0H, C1H, C2H, C3H, C4H, C5H, C6H, C7H, C8H, C9H, CAH, CBH, CCH, CDH, CEH, CFH in a page mode fashion respectively.

Once the all 16 data values are written, the data at addresses C2 and C4 are read back. The I/O description of this DRAM is shown below.

![Diagram of DRAM I/O](image)

(a) Draw the timing diagram for the page mode write followed by the 2 single reads described above. How many addresses are in the address space of the system if only this DRAM chip is used?

(b) If 8 such DRAMs are used to store 32 bit of data instead of 4, what would be the total address space of the new system if Little Endian memory format is used? Draw the schematic of this memory group.
25 pts

Address

RAS

CAS

DIn

Dout

WE

Since the effective address is 8 bits wide, the address space is 256.

25 pts

(b)

Address space is still 256.

Since the effective address is 8 bits wide, the address space is 256.
An interrupt controller which services 4 hardware interrupts is connected to a 16-bit CPU, which consists of 16-bit wide instruction (Imem) and data (Dmem) memories, program (PC) and data (DC) counters, A and B registers and a CPU controller.

This schematic is shown below:

The interrupt protocol in this schematic works as follows:

**Step 1:** Active high interrupt, INTR, is generated by the interrupt controller for CPU because of one or more hardware interrupts. Interrupt controller must have all the active hardware requests from external devices with interrupt IDs before generating INTR.

**Step 2:** CPU acknowledges with an active high interrupt acknowledge, INTA.
Step 3: Interrupt controller produces an interrupt ID on the 16-bit bi-directional data bus (the choice of the interrupt ID is determined by the priority unit in the interrupt controller).

Step 4: The interrupt ID is loaded to the PC to access the interrupt service subroutine address.

Step 5: The interrupt service subroutine address is loaded to the PC.

Step 6: The 4 consecutive interrupt service instructions are fetched from the instruction memory at the PC address for the particular interrupt. These steps are as follows:

Step 6a: The first interrupt instruction specifies the data memory address to be loaded to the DC.

Step 6b: The second instruction contains the value of A to be loaded to the A-register.

Step 6c: The third instruction contains the value of B to be loaded to the B-register.

Step 6d: The fourth instruction provides either the results of ADD or the contents of A-register or the contents of B-register to be loaded to the data memory at the DC address; this value will later be used in the program (the related hardware is not shown in the schematic above).

Step 7: When the 4-cycle interrupt service is complete, CPU lowers INTA; the interrupt controller lowers INTR a cycle after INTA and gets ready for the next interrupt.

Note: Imem or Dmem are SRAM memories; storing data is achieved with WE within the same clock cycle; loading data is achieved with RE with a latency of 1 clock cycle; OE enables/disables memory’s tri-state buffer to access to the data bus.

Question 1: If the priority scheme in the interrupt controller is such that device 0 has the highest and device 3 has the lowest priority, design this controller in gate-level with the I/O port description shown in the schematic. Note that this controller supports only hardware interrupts (30 points).

Question 2: Show the timing diagram of the complete interrupt service outlined from step 1 to step 7 above (40 points).

Question 3: Show the CPU controller state diagram by including all the control inputs to operate instruction and data memories, registers etc. (30 points).
<table>
<thead>
<tr>
<th>Req3</th>
<th>Req2</th>
<th>Req1</th>
<th>Req0</th>
<th>ID(8bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No ID</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Int0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Int1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Int0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Int1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Int0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Int3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Int1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Int0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Int2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Int0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Int0</td>
</tr>
</tbody>
</table>

Int0 = Req0
Int1 = Req0 Req1
Int2 = Req0 Req1 Req2
Int3 = Req0 Req1 Req2 Req3

Diagram of the circuit with inputs and outputs.
This system contains 3 bus masters, MPU, its co-processor and DMA, and 3 slaves, Memory I, Memory 2 and Memory 3.

Diagram:

Question 1: A 16-bit system with unidirectional data and address buses is given below.

Dr. Amrit Bindal
value of address or data in the appropriate box (30 points). Note that if any control signal is at logic 0, insert 1 in the appropriate box in this table. Similarly, write the DMA memory transfers using the table on the next page.

Including the DMA requests, Grant, data, address and the control signals provided for each memory, create the timing diagram of the cycle. DMA has no temporary register to store intermediate data. DMA follows a bus protocol in which memory address and controls are produced at the first cycle; data is dispatched in the next clock cycle; while the clock cycle is dispatched within the same clock as dispatching the address. Both memories have a read access time of 1 clock cycle: DMA to Memory 1 works at a clock frequency twice as high as Memory 2.

Note that, DMA and Memory 1 works at a clock frequency twice as high as Memory 2.

ABCD of memory 1, respectively, and 1000 and 0000 of memory 2, respectively; the second one is from address 0003 of memory 2 and ABCE and 1000 of memory 1, respectively; the first one is from address ABCD and ABCE of memory 1 and its co-processor are at idle. DMA requests two data transfers.

(a) Write MPU and its co-processor are at idle, DMA requests two data transfers. (b) While MPU and its co-processor are at idle, DMA requests two data transfers.

1) When the description above, draw the state diagram of the controller for the bus arbitration (40 points).

2) With the description above, draw the state diagram of the controller for the bus arbitration (40 points). Note that the control signals are not shown to avoid the complexity on the schematic; however, each memory has Read Enable (RE) and Write Enable (WE) ports to control the data storage.

Similarly, the bus master can read data from the slave via a 16-bit-wide data (RD/WR) bus. As long as a bus master owns the bus, it can send its 16-bit-wide write data (W/D) and address (ADD) to the selected slave.

Priority list:
- For the memory is high, when the request signal goes low, the arbiter lowers its Grant and assigns the bus to the pending request in the lowest priority to use the bus. When the request from a bus master is granted, the bus master owns the bus as long as its request signal is high. The arbiter is responsible for assigning the usage of the bus among the 3 bus masters. MPU has the highest and the DMA has the...
According to the diagram, draw the schematic for an 8-bit timer (please do not draw any logic diagrams). Make sure all the I/O ports are labeled clearly in this diagram (30 points).

The timer has also an asynchronous active low reset. Before the program starts, the user resets the timer.

Question 2: The timing diagram for a simple 8-bit timer is given below.
Diagram of a state transition diagram with states and transitions labeled with conditions and actions. The states include IDLE, Grantm, Grantc, and Grantd, with transitions marked by arrows and conditions such as reqm, reqc, and reqd.
DataIn 8

select port 2 if WE = 1
select port 1 if Gate = 1

reset

clk

Output

DataOut
Question 1 (Fundamentals)

(a) A module below creates the following timing diagram. The module has an active high reset and a single output. Design and draw the circuit schematic with basic gates, mega cells (muxes, adders etc.) to generate this timing diagram. Do not use any state diagram in your solution (25 points).

(b) A similar module in (a) creates the following timing diagram. The module has again an active high reset and a single output. Design and draw the circuit schematic with basic gates, mega cells (muxes, adders etc.) to generate this timing diagram. Do not use any state diagram in your solution (25 points).
Question 2 (Bus transfer)

A bus master writes 4 packets of data (bytes) at the following address locations of a 32-bit wide byte-addressable memory (slave) organized in a Little Endian format:

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0DH</td>
<td>11H</td>
</tr>
<tr>
<td>11H</td>
<td>22H</td>
</tr>
<tr>
<td>15H</td>
<td>33H</td>
</tr>
<tr>
<td>19H</td>
<td>44H</td>
</tr>
</tbody>
</table>

Following the write cycle, the same bus master reads data (words) from the following addresses of the same slave:

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>3CH</td>
<td>AABBCDDH</td>
</tr>
<tr>
<td>40H</td>
<td>55667788H</td>
</tr>
</tbody>
</table>

(a) Draw the memory contents after writing and reading take place (10 points).

(b) If the first address is generated at clock cycle #1 by the bus master when slave produces the "ready" signal as in the timing diagram below, fill in the address, control and data entries in this timing diagram (40 points).
1. SDRAM operation (50 points):

Two READs can be accomplished from a 16-bit wide SDRAM by a single CPU instruction. SDRAM is organized with 4 banks with data shown below.

<table>
<thead>
<tr>
<th>Bank0</th>
<th>Bank1</th>
<th>Bank2</th>
<th>Bank3</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA00</td>
<td>BB00</td>
<td>CC00</td>
<td>DD00</td>
</tr>
<tr>
<td>1111</td>
<td>AAAA</td>
<td>FFFF</td>
<td>8888</td>
</tr>
<tr>
<td>AA01</td>
<td>BB01</td>
<td>CC01</td>
<td>DD01</td>
</tr>
<tr>
<td>2222</td>
<td>BBBB</td>
<td>EEEE</td>
<td>7777</td>
</tr>
<tr>
<td>AA02</td>
<td>BB02</td>
<td>CC02</td>
<td>DD02</td>
</tr>
<tr>
<td>3333</td>
<td>CCCC</td>
<td>DDDD</td>
<td>6666</td>
</tr>
<tr>
<td>AA03</td>
<td>BB03</td>
<td>CC03</td>
<td>DD03</td>
</tr>
<tr>
<td>4444</td>
<td>DDDD</td>
<td>CCCC</td>
<td>5555</td>
</tr>
<tr>
<td>AA04</td>
<td>BB04</td>
<td>CC04</td>
<td>DD04</td>
</tr>
<tr>
<td>5555</td>
<td>EEEE</td>
<td>BBBB</td>
<td>4444</td>
</tr>
<tr>
<td>AA05</td>
<td>BB05</td>
<td>CC05</td>
<td>DD05</td>
</tr>
<tr>
<td>6666</td>
<td>FFFF</td>
<td>AAAA</td>
<td>3333</td>
</tr>
<tr>
<td>AA06</td>
<td>BB06</td>
<td>CC06</td>
<td>DD06</td>
</tr>
<tr>
<td>7777</td>
<td>0000</td>
<td>1111</td>
<td>2222</td>
</tr>
<tr>
<td>AA07</td>
<td>BB07</td>
<td>CC07</td>
<td>DD07</td>
</tr>
<tr>
<td>8888</td>
<td>1111</td>
<td>0000</td>
<td>1111</td>
</tr>
</tbody>
</table>

Each SDRAM address is composed of an 8-bit wide row address, RA[7:0], and an 8-bit wide column address, CA[7:0], as in the following format:

**SDRAM Address = \{RA[7:0], CA[7:0]\}** where the row address occupy higher bits.

To control SDRAM, one has to supply the following controls at the negative edge of the system clock:

<table>
<thead>
<tr>
<th>Operation</th>
<th>CS</th>
<th>RAS</th>
<th>CAS</th>
<th>WE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precharge</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Activate</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Read</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

The wait period between precharge cycle and activate is 3 cycles, between activate and read is 2 cycles. The precharge cycle for the next read operation takes place after the last data of the burst is read out from SDRAM.

Note that BS[1:0] = 0 selects Bank0, BS[1:0] = 1 selects Bank1, BS[1:0] = 2 selects Bank2 and BS[1:0] = 3 selects Bank3 in the timing diagrams.
Part (a) (20 points)
Assuming that the mode register is pre-programmed with a burst length of 4, a CAS latency of 2 and sequential mode addressing before any CPU read instruction is executed, fill the appropriate entries in the timing diagram below such that the two reads are from SDRAM Addresses AA00 and AA04. Start from the precharge cycle to accomplish each read.
Part (b) (20 points)
With the same mode register contents in part (a), fill the appropriate entries in the timing diagram below such that the two reads take place in shortest possible time and they are from the SDRAM Addresses AA00 and BB02. Again start from the precharge cycle to accomplish each read.
Part (c) (10 points)
With the same mode register contents in part (a), accomplish one read from SDRAM Address CC00 with a burst length of 2 and one read from SDRAM Address DD02 with a burst length of 8. Start from the precharge cycle to accomplish each read.
2. Transmitter/Receiver Operation (50 points)

A data exchange takes place between a transmitter and a receiver with Req and Ack handshake control signals as shown below. In this block diagram, transmitter clock frequency is twice as high as the receiver clock frequency. Positive edges of transmitter and receiver clocks are aligned with each other.

The handshake protocol between the transmitter and receiver is as follows:

When the receiver requests data, it sends its request signal at the positive edge of the receiver clock. When the transmitter receives the request, it acknowledges at the positive edge of the transmitter clock and starts streaming data. At the positive transmitter clock edge when the last data is sent, the transmitter lowers its acknowledge signal; the receiver, in turn, sees transmitter’s response and also lowers its request signal at the positive edge of its clock.

Assuming that the transmitter delivers 4 data packets in one burst, does the receiver receive all these data? How do the request and acknowledge signals change with each other? Show all these using a timing diagram where clockRx waveform is plotted at the top row, ReqRx at the second, clockTx at the third, AckTx at the fourth and Data at the bottom row.

DO NOT USE THE QUESTION SHEET FOR SOLUTIONS!!
data constant after the burst

receivers can only read DI & D3 from the transmitter (D4 also in TX holds this)

[Diagram of waveforms and signals]
**Question 1 (DMA)**

A simple DMA is connected to source and destination memories with an 8-bit wide “multiplexed”, bidirectional bus. Because this bus is multiplexed, source memory address, destination memory address and data have to be carried by the same bus at different times. Both source and destination memories have WE, RE, AddrIn and DataIn inputs, Ready and DataOut outputs. When the Ready signal is issued by memory, DMA can generate an address in the next cycle. As soon as a valid address is supplied to a memory, an 8-bit data is read from memory in the next cycle (latency = 1).

Part a (25 points)

Considering the functionality described above, show a micro-architectural block diagram that contains DMA, source and destination memories and the appropriate control signals for each unit. Because of the nature of the multiplexed bus, you will need other modules besides the DMA and the memories. Include these blocks with their control inputs to your diagram.

Part b (25 points)

Now, generate a sample timing diagram including Address/Data, ReadyS, RES, WES (where S stands for source), ReadyD, RED, WED (where D stands for destination) and other control signals you may need for proper data flow. Show the I/O ports of your controller and design its functionality using Moore-type state diagram. Ignore any address counter(s) in the DMA unit to simplify your design.

Please label your solution as “Question 1 Part a” or “Question 1 Part b”. Unmarked solutions will not be considered.

PLEASE DO NOT WRITE THE REST OF THIS PAGE!
(b) Sample timing diagram is as follows:

```
clk
A/D
ReadyS
ReadyD
RES
nWEBufS
nWE
OE_DMA
OE_S

A1S  DATA1  A1D

A2S  DATA2  A2D
```
From the sample timing diagram, the Moore state machine of the controller is as follows:

\[
\begin{align*}
\text{Readys} &= 0 \\
\text{IDLE} \\
\text{Readys} &= 1 \\
\text{Address} \\
\text{RES} &= 1 \\
\text{OEDMA} &= 1 \\
\text{Ready D} &= 0 \\
\text{WEB} &= 1 \\
\text{OES} &= 1 \\
\text{Ready D} &= 1 \\
\text{Readys} &= 0 \\
\text{Address} \\
\text{WED} &= 1 \\
\text{OEDMA} &= 1
\end{align*}
\]
**Question 2 (CPU)**

A 16-bit RISC CPU organized in Big Endian format executes OUT, IN, ADD and SUB in 4 stages. This CPU has separate instruction and data memories, a register file with 16 registers. In the first clock cycle, instruction is fetched from the instruction memory. In the second cycle, register file is accessed to read operand contents. The third cycle uses ALU and/or data memory. In the fourth cycle, either write to the register file or external memory is completed.

OUT is an instruction that moves data from CPU’s data memory to an external memory using system bus (no DMA assistance is needed)

\[
\text{OUT RS, Imm, RD } \text{MemData[Reg[RS] + imm} \rightarrow \text{MemExt[Reg[RD]]}
\]

\[
\begin{array}{cccccc}
0 & 12 & 45 & 78 & 15 \\
\text{OUT} & \text{RS} & \text{RD} & \text{Imm}
\end{array}
\]

IN is an instruction that moves data from external memory to CPU’s data memory using system bus (no DMA assistance is needed)

\[
\text{IN RS, Imm, RD } \text{MemExt[Reg[RS]]} \rightarrow \text{MemD[Imm + Reg[RD]]}
\]

\[
\begin{array}{cccccc}
0 & 12 & 45 & 78 & 15 \\
\text{IN} & \text{RS} & \text{RD} & \text{Imm}
\end{array}
\]

ADD is an instruction that adds two operand contents and stores the result into the register file

\[
\text{ADD RS1, RS2, RD } \text{Reg[RS1] + Reg[RS2]} \rightarrow \text{Reg[RD]}
\]

\[
\begin{array}{ccccccc}
0 & 12 & 45 & 78 & 10 & 11 & 15 \\
\text{ADD} & \text{RS1} & \text{RS2} & \text{RD} & \text{Not used}
\end{array}
\]

SUB is an instruction that subtracts two operand contents from each other and stores the result into the register file

\[
\text{SUB RS1, RS2, RD } \text{Reg[RS1] - Reg[RS2]} \rightarrow \text{Reg[RD]}
\]

\[
\begin{array}{ccccccc}
0 & 12 & 45 & 78 & 10 & 11 & 15 \\
\text{SUB} & \text{RS1} & \text{RS2} & \text{RD} & \text{Not used}
\end{array}
\]

Show the schematic of this system including the external memory, CPU and its internal memories, register file, detailed ALU and other glue logic. Illustrate all bus widths, label all data, address and control signals for the proper data flow.
Question 1 (50 points)

A bus master reads 4 data packets starting from address 00 until address 03 from an 8-bit wide memory. This is immediately followed by writing 0x00, 0x11, 0x22 and 0x33 into addresses 04, 05, 06 and 07 respectively. This memory contains the following data after this operation:

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0xAA</td>
</tr>
<tr>
<td>01</td>
<td>0xBB</td>
</tr>
<tr>
<td>02</td>
<td>0xCC</td>
</tr>
<tr>
<td>03</td>
<td>0xDD</td>
</tr>
<tr>
<td>04</td>
<td>0x00</td>
</tr>
<tr>
<td>05</td>
<td>0x11</td>
</tr>
<tr>
<td>06</td>
<td>0x22</td>
</tr>
<tr>
<td>07</td>
<td>0x33</td>
</tr>
</tbody>
</table>

Assuming this is a unidirectional bus with separate RData and WData buses, fill in the empty spaces in the timing diagram below:
Question 2 (50 points)

A bus master is connected to 4 slaves (memories) in a bidirectional bus where 32-bit address, write and read data are transferred sequentially on the same bus. The I/O ports of the bus master and the slaves are shown below:

<table>
<thead>
<tr>
<th>Bus Master</th>
<th>Bus Slave</th>
</tr>
</thead>
<tbody>
<tr>
<td>RData [31:0]</td>
<td>Addr/Data [31:0]</td>
</tr>
<tr>
<td>WData [31:0]</td>
<td>EN</td>
</tr>
<tr>
<td>Addr [31:0]</td>
<td>W/R</td>
</tr>
<tr>
<td>EN</td>
<td></td>
</tr>
<tr>
<td>W/R</td>
<td></td>
</tr>
<tr>
<td>Ready</td>
<td>Ready</td>
</tr>
</tbody>
</table>

The bus master has separate read, write data and address ports to receive/transmit address and data. The Enable (EN) = 1 and W/R = 1 ports enables the bus master to write; EN = 1 and W/R = 0 enables it to read. The bus master is aware of slave’s readiness to receive/transmit data through Ready port.

The slave, on the other hand, has only one port to receive address and data. Active high Enable (EN) and Write Enable (WE) ports on the slave are used to activate the slave and writing data into the slave, respectively. If data needs to be read from the slave, active high EN is still used along with WE = 0.

Draw the architectural diagram of such a system. Make sure to use more significant address bits emerging from the bus master to select one of the slaves for the bus master to read or write data.
Question 1 (50 points):

A DMA controller is transferring 4 words of data, D1, D2, D3 and D4, from SRAM memory 1 (source memory) to SRAM memory 2 (destination memory) on a 32-bit wide bidirectional bus as shown below. The address values of D1, D2, D3 and D4 are fetched from AS1, AS2, AS3 and AS4 in the source memory and placed at AD1, AD2, AD3 and AD4 in the destination memory, respectively. Since each SRAM memory has a single port, address and data cannot exist in the same clock cycle. Therefore, for read operation, data becomes available at the SRAM port the cycle after a valid address is presented. For write operation, on the other hand, data has to be presented to the SRAM port the cycle after the address. Active-high RE and WE enables the SRAM read or write operations when a valid address is introduced. The DMA controller has 2 programming ports to program the initial address and the incremented value using ProgAddrS (ProgAddrD) and ProgIncrS (ProgIncrD) for the source and destination address pointers. The active-high StartS (StartD) pin produces the first address at AddrOutS (AddrOutD); the incremented addresses are produced by the active-high IncrS (IncrD) pin.

Part A (10 points)
Draw the schematic for the source and destination address pointers.

Part B (40 points)
Complete the timing diagram on the next page to transfer 4 words of data from memory 1 to memory 2.
DON'T WRITE ANYTHING IN THIS SPACE!
Question 2 (50 points):

**Part A (25 points)**
A rate generating timer produces active high pulse every 128 cycles as shown below. Design this timer and draw its schematic.

**Part B (25 points)**
A one-shot timer produces a continuous pulse 100 cycles after the start signal as shown below. Design this timer and draw its schematic.

DON'T WRITE ANYTHING IN THIS SPACE!
Part A: Rate generator

Clock

CountOut: 0 1 2 3

Out

Diagram:

+1

\[ \begin{array}{c}
\text{CountOut} \quad \text{0} \quad \text{1} \quad \text{2} \quad \text{3} \\
\text{Out} \\
\end{array} \]
Part B: One-shot timer

clock
Start
CountOut
Out

\[ +1 \]
Start

\[ \text{Out} \]