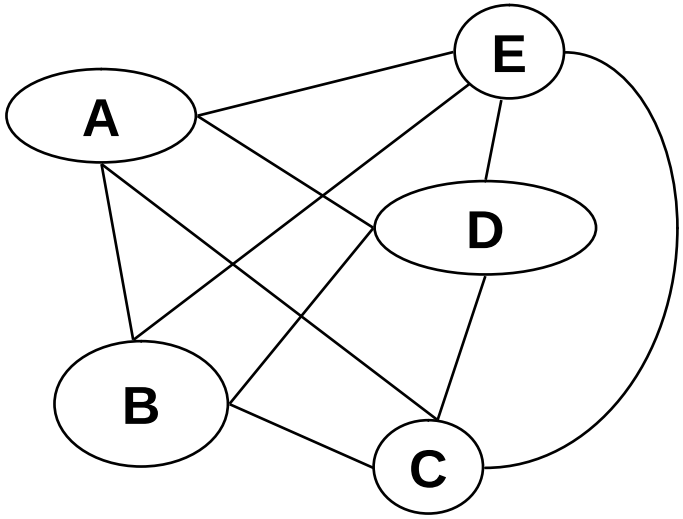
Homework

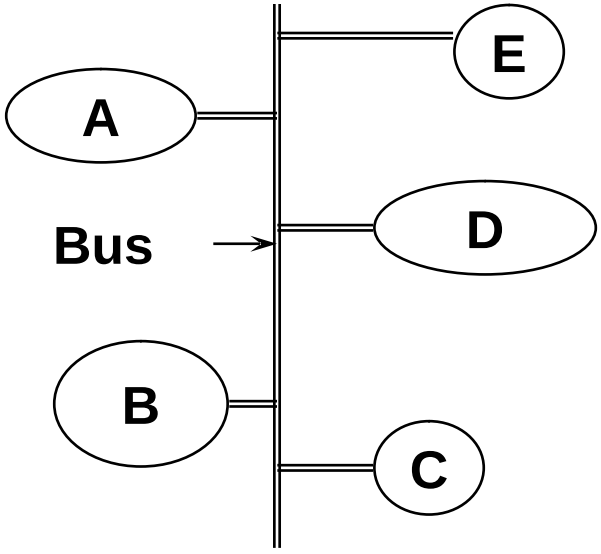
- Reading
  - Tokheim, Section 13-5 (Three State Buffers only)
- Machine Projects
  - Continue on MP5
- Labs
  - Continue labs with your assigned section

Buses

- Concept is to link together multiple functional units over a common data highway at a lower cost than using multiple point to point links

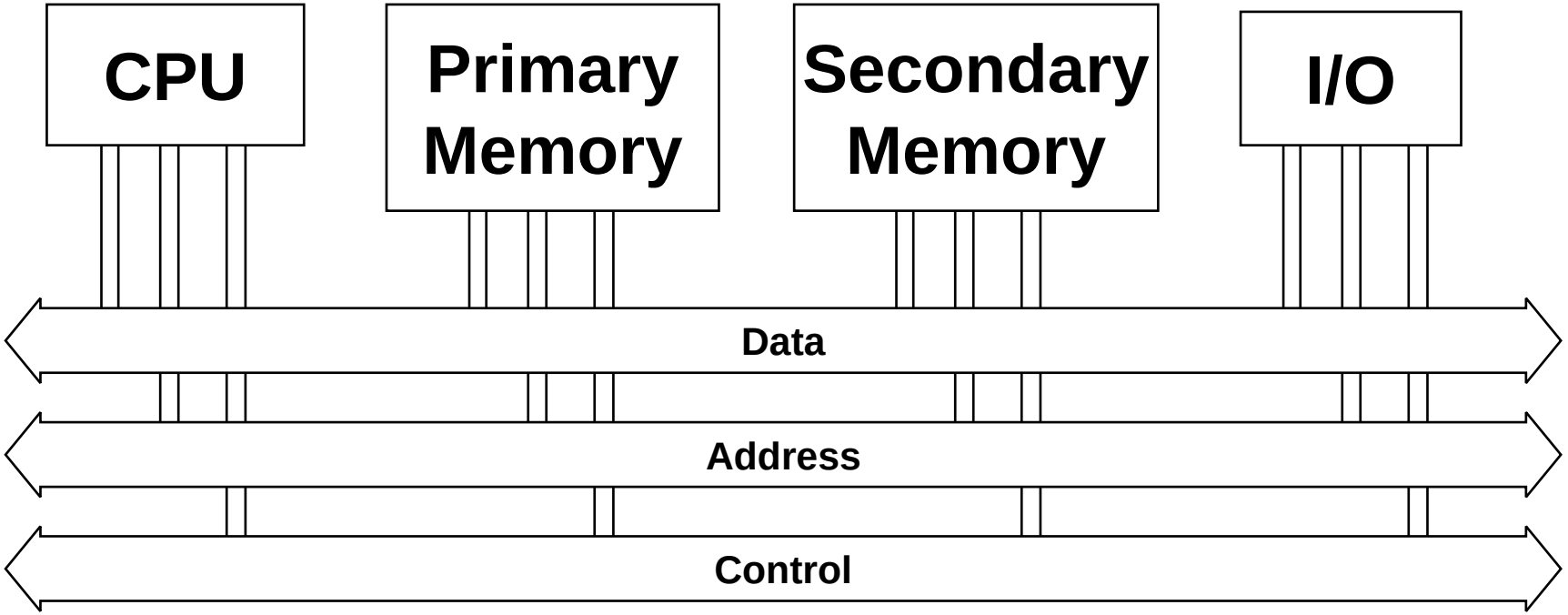


OR



Number of Links = n * (n – 1) / 2

Bus - Essential Part of Any Computer



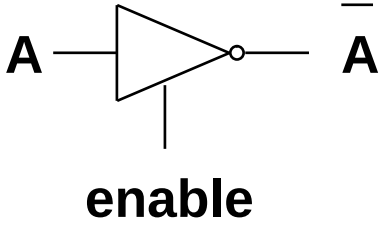
Tri-state Logic Outputs

- Since we can have multiple masters on a bus, we need Tri-state logic for attachment to a bus so that each device can choose to drive or not drive the bus depending on whether it is the bus master for a given bus cycle
- Tri-state logic prevents a bus conflict where one device is driving a signal to 1 and another device is driving it to 0 at the same time - generates high current through wires (and smoke?)

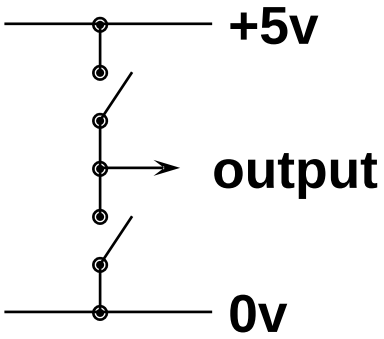
Tri-State Logic

- The problem with connecting multiple “normal” outputs together on a bus is that each has to be in one logic state (0) or the other (1) - driving voltage on each bus signal high or low
- This represents a conflict over the state of the signal
- We resolve this conflict with *tri-state logic*

Logically



Electrically

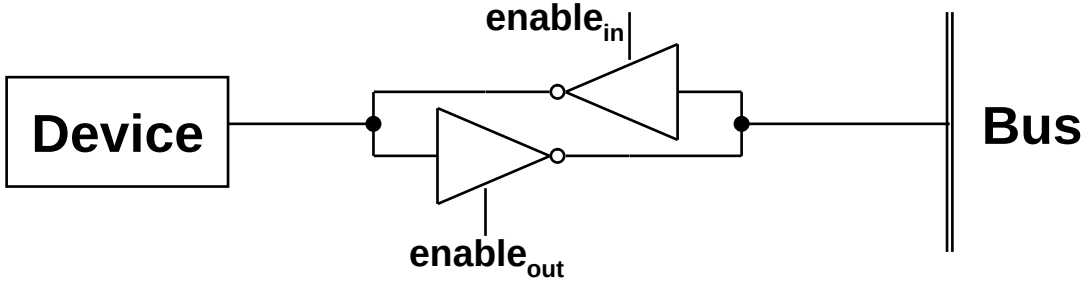


Truth Table

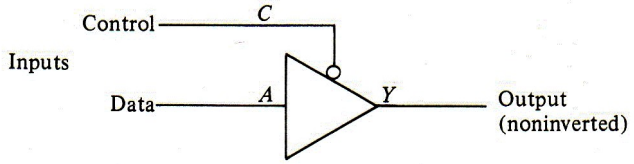
enable	A	Output
0	0	(Z)
0	1	(Z)
1	0	1
1	1	0

Tri-State Logic and Buses

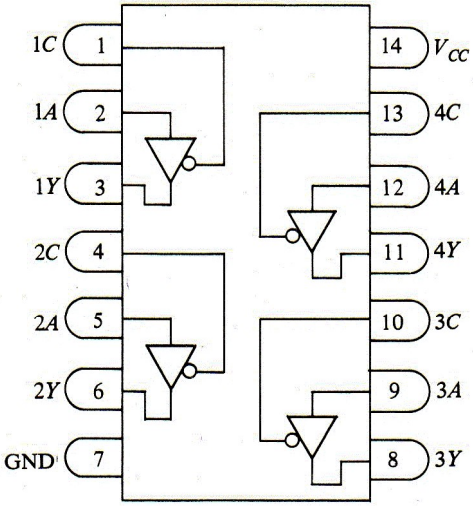
- The logical element has output enable pin to go from a floating output to drive the output from the circuit
- Inverters and buffers are used as bus drivers or buffers
  - Two such drivers or buffers in opposite directions are used to make the connection bi-directional
  - The gates also provide more “drive” onto the bus so that the bus signals are stronger and the bus can be longer



Tri-State Logic and Buses



(*a*) Logic symbol of a three-state buffer



(*b*) Pin diagram

Inputs		Output
C	*A*	*Y*
L	L	L
L	H	H
H	X	(Z)

L = LOW voltage level
H = HIGH voltage level
X = don't care
(Z) = high impedance (off)

(*c*) Truth table

Fig. 13-12 74125 quad three-state buffer IC

Bus Master – Slave Relationships

- During any specific bus cycle, only one device attached to the bus is allowed to drive it
- Driving the bus means that a device is forcing each signal on the bus to a high or low state
- For the data bus, the processor, a memory chip, or an I/O device may be driving the data bus during a specific read or write bus cycle
- Specific signals on the address and control bus select a device to be the master on the data bus

Bus Master – Slave Relationships

- Up till now, I have said that the address bus and the control bus are always driven by the processor, however that is NOT really true!
- That was only a “lie of simplification”!
- The processor is NOT the only device that may be driving the address and control busses
- Hopefully you are now well-prepared for me to un-simplify a bit J

Bus Arbitration

- Bus arbitration is used to hand off a bus between one of several potential bus masters using signals that are a part of the bus itself
- A bus arbitration protocol implements some form of bus request and bus grant handshake to determine which device will be the master on the bus for the next bus cycle

Bus Master – Slave Relationships

- Other devices that potentially can be the master on the address and control bus are:
  - Direct Memory Access (DMA) Controller
  - DRAM controller to refresh the stored bits
  - Other processors in multiprocessor architectures
- We'll only discuss the first application above, but not at the level of programming for it in detail – too hard to un-simplify that

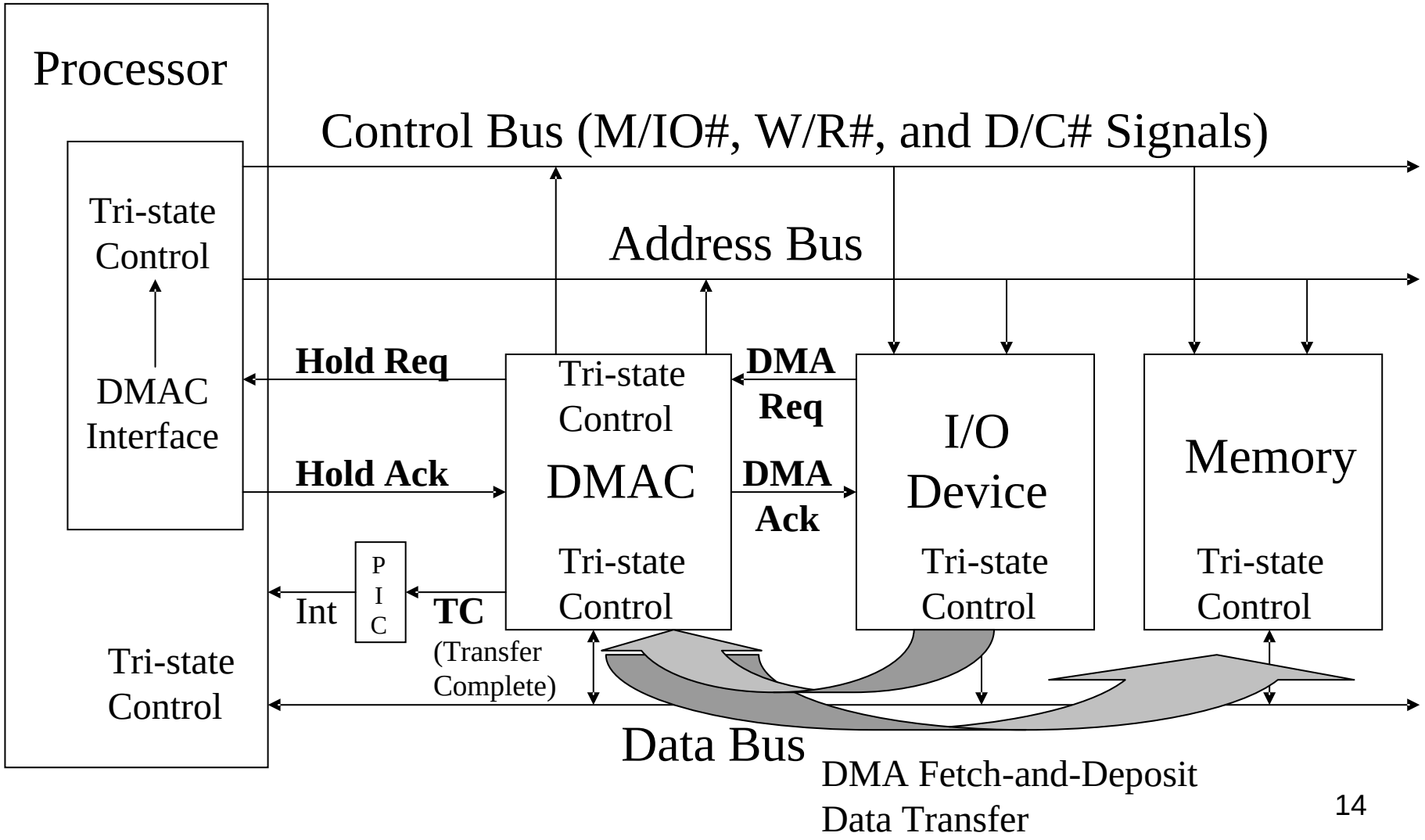
Direct Memory Access

- You thought that handling an I/O device under interrupt control was pretty good – right?
- Wrong!
- The overhead to process an interrupt for each byte of data is still relatively costly in terms of processor time
  - Process interrupt and stack context of processor
  - Fetch and execute instructions of ISR
  - Move data from I/O device or memory to processor register
  - Move data from processor register to memory or I/O device
  - Restore context of processor and resume background code
- It is better if an I/O device like a disk can transfer data directly to or from the memory without the processor needing to execute any in or out instructions!

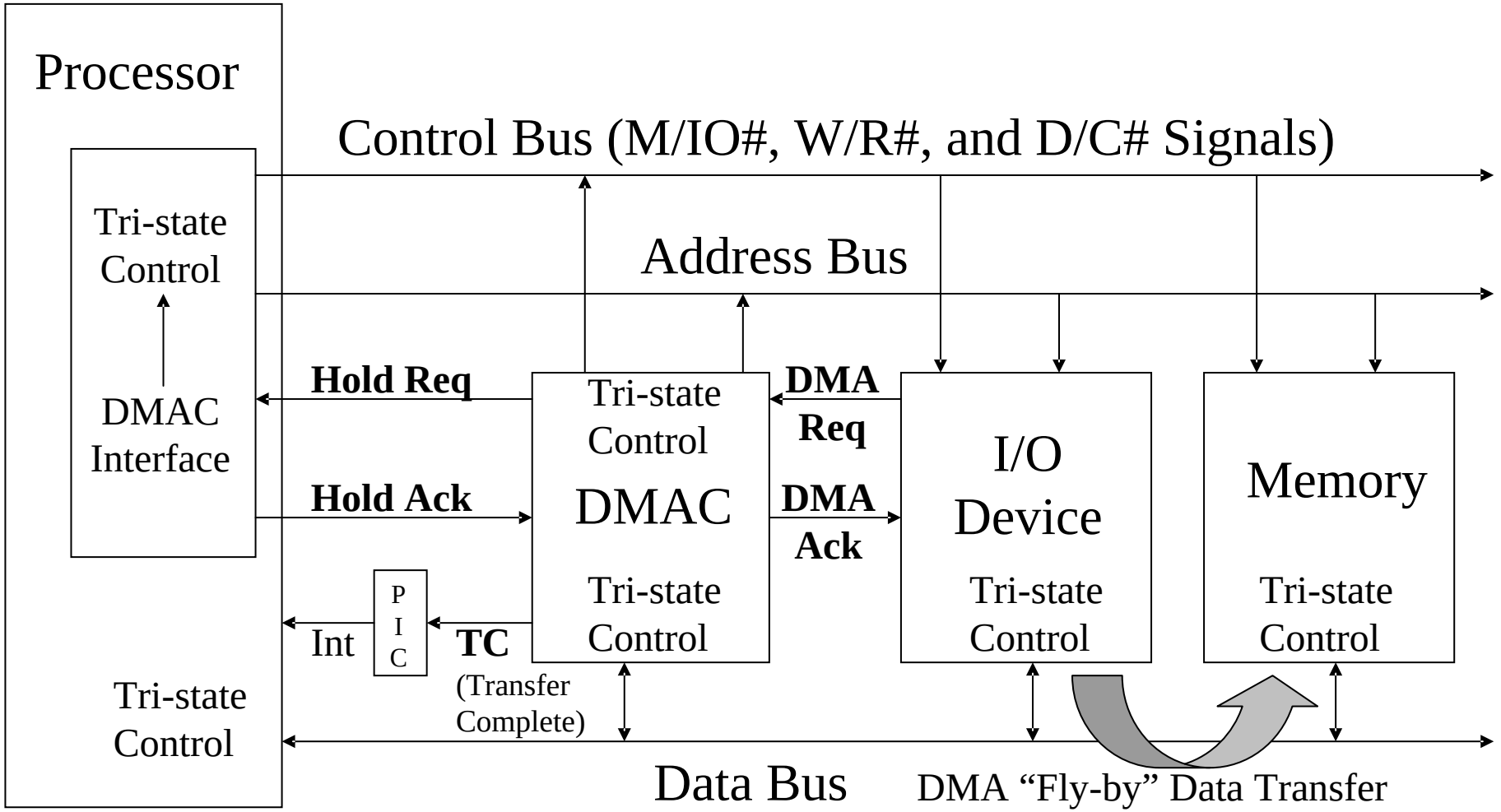
Direct Memory Access

- We add a DMA Controller (DMAC) to our system, e.g. Intel 8237A DMA controller chip
- The DMAC has the capability of becoming the bus master on the address and control bus for one or more “channels” transferring data between an I/O device and memory, e.g. 8237A supports 4 channels
- We connect DMA Request and DMA Acknowledge signals between the I/O device and the DMAC
- Software in the CPU sets up the DMAC to transfer an entire sequence of bytes between memory buffer and the I/O device or vice versa
- During each byte transfer, the DMAC drives address and control bus signals instead of the processor

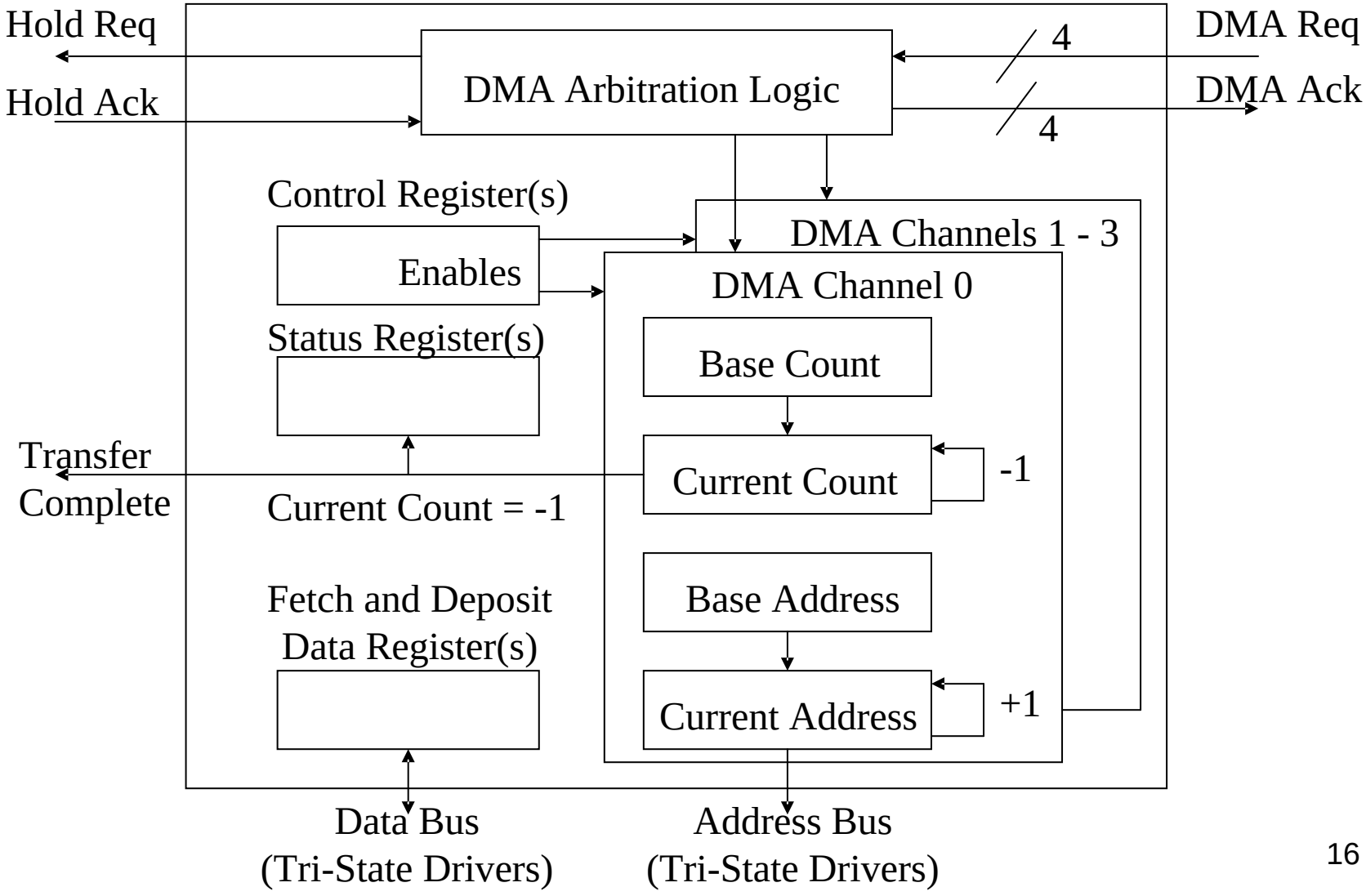
Direct Memory Access



Direct Memory Access



DMA Controller Details



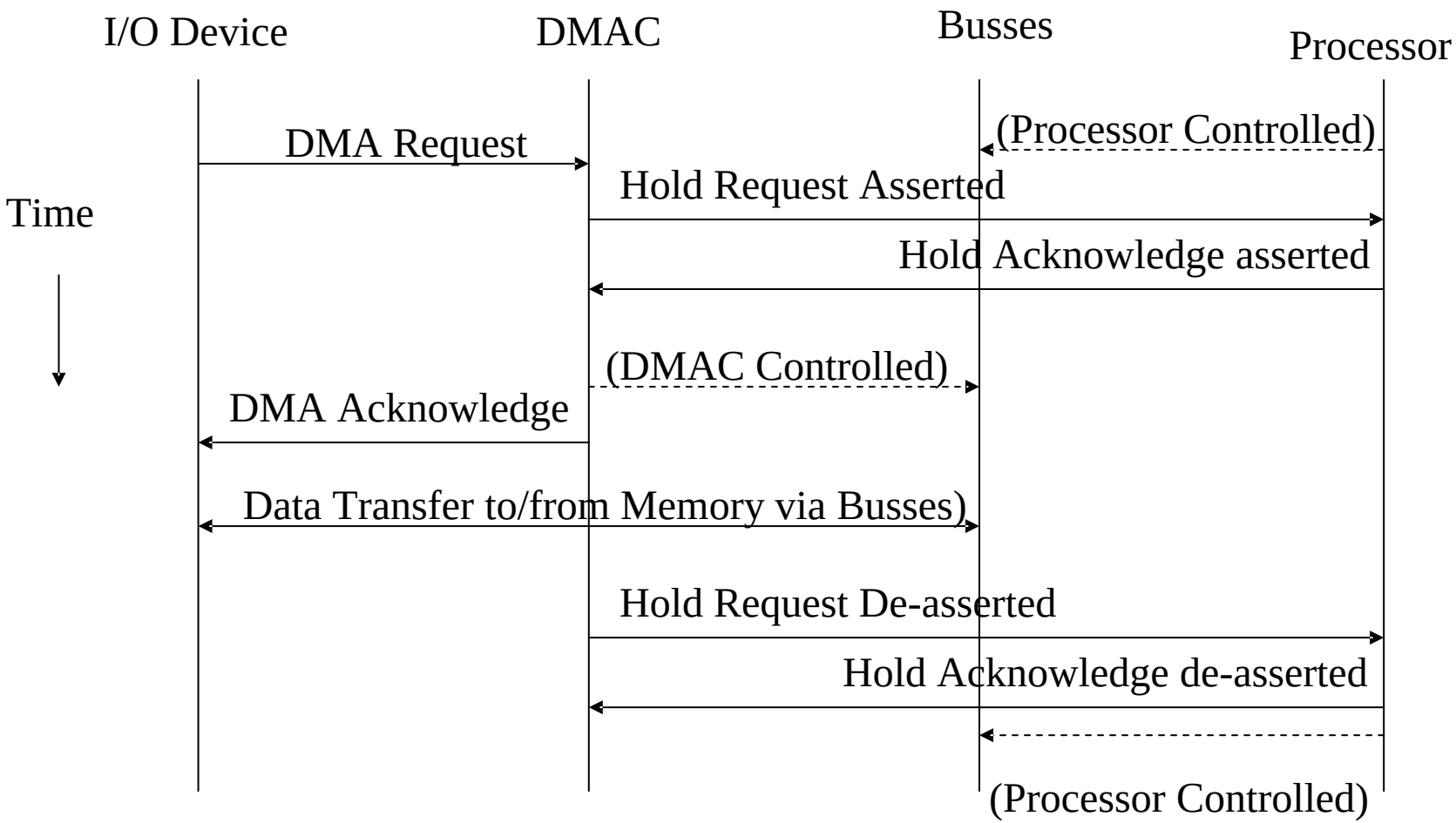
DMAC Programming

- Generalized Steps for Programming DMAC
  - Allocate a suitably-sized memory buffer
  - Disable the DMA channel being programmed (Note that cli does not stop DMA cycle stealing)
  - Set the Base Address Register with buffer address
  - Set the Base Count Register with size of transfer
  - Set the DMA transfer mode
  - Enable the DMA channel to start the transfer

DMAC Operation

- When requested, the DMAC arbitrates with the CPU to be the master on the address and control busses
- It executes a bus cycle to transfer a byte of data from memory (or I/O device) to I/O device (or memory)
- While DMA controller is bus master, the CPU can not access memory or I/O devices
- This is called “Cycle Stealing” (the DMA controller steals bus cycles from the processor)
- It takes less time than executing an ISR for each byte

DMAC Bus Arbitration



DMAC Operation

- When the DMAC finishes transferring an entire block of data between I/O device and memory
  - It interrupts the processor (via TC to the PIC)
  - ISR software in the processor sets up DMAC again for transferring the next block of data
- Processor gets interrupted for I/O handling much less often than once per byte of data
- Hence, much better processor performance

Tri-State Bus Summary

- All devices have tri-state logic connections to the data bus – may be driving or receiving
- Memory and I/O devices don’t need tri-state logic on address/control bus (never drive them)
- Because the processor may need to yield the control/address busses, it must have tri-state logic for driving those bus signals
- DMAC controller must have tri-state logic for driving the control and address bus signals