Registers
A 4-Bit Register

Could be called a parallel-in/parallel-out register.

Why?
A Shift Register

Called a serial-in, parallel-out shift register (SIPO)
SIPO Register (Serial-In/Parallel-Out)
SISO Register (Serial-In/Serial-Out)

Useful for *delaying* a serial bit-stream some number of cycles...
How to Make Any of These Loadable?

This register loads on every clock cycle. How to make it load when told to?

An obvious solution...

This is incorrect - why?
**Gated Clocking** Is A Bad Thing To Do…

The flip flop gets its clock signal late. Thus, its output appears late…

The flip flop gets its clock signal late.

Thus, its output appears late…
Gated Clocking

• Different flip flops load at different times
  - A form of *clock skew*
  - Makes doing timing analysis more difficult
  - Can lead to circuits which *run* more slowly
  - Can lead to circuits which *fail* at any clock rate
Value that should be loaded into $FFB$ ('1')

Value that gets loaded into $FFB$ ('0')
If $t_{CLK-Q} < t_{AND}$:
FFB loads wrong value

Value that should be loaded into FFB ('1')
Value that gets loaded into FFB ('0')
Globally Synchronous Design

• One global clock
• All registers load on that clock’s edge
• Control over loading done via input forming logic (IFL)

• Simplifies timing analysis and requirements
• Makes it possible for even novices to design large, functioning circuits

• Multi-clock circuits ⇔ next semester’s topic
The Correct Way To Make A Loadable Register (1-Bit)

When LOAD='0', FF loads old value

When LOAD='1', FF loads DIN
A Correct Scenario

No clock skew
Both FF’s load on exactly same clock edge
Both Q’s appear at same time
Timing analysis greatly simplified

$$T_{cycle} \geq T_{CLK-Q} + T_{IFL} + T_{SETUP}$$

$$T_{cycle} \geq T_{CLK-Q} + MAX \left\{ \frac{T_{NOT}}{T_{MUX21}} \right\} + T_{SETUP}$$
A Loadable Parallel-In, Parallel-Out Register

PIPO?
A Loadable Parallel-In, Serial-Out Register (PISO)

When LOAD/SHIFT# = ‘1’, register loads D2-D0

When LOAD/SHIFT# = ‘0’, register shifts right

This register is always either loading or shifting...
MUX for Register Control

- Loadable register concept can be generalized
  - Provide any combination of inputs to register
A SISO With An Enable Input

Combination of loadable register with shift register...
When ENABLE=‘0’, register doesn’t shift (re-loads old value)
When ENABLE=‘1’, register shifts
A Bidirectional Shift Register

Top-In

Q1

Q2

UP/DOWN#

CLK

Bottom-In

Q1

Q0

Q2

UP/DOWN#

CLK

ECE238L
Uses of Shift Registers

• Collecting serial input data into a parallel word
• Shifting out bits of a word
• Delaying a serial stream by some # of cycles
A Clearable Counter

From there to here, from here to there, interesting circuits are everywhere...

(when you have a MUX and some flip flops)
An Up/Down Counter

How about an up/down counter + bi-directional shift register design?
Up/Down Counter + Bi-Directional Shift Register

<table>
<thead>
<tr>
<th>Control</th>
<th>NextQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Q+1</td>
</tr>
<tr>
<td>01</td>
<td>Q-1</td>
</tr>
<tr>
<td>10</td>
<td>Q shifted left</td>
</tr>
<tr>
<td>11</td>
<td>Q shifted right</td>
</tr>
</tbody>
</table>

Control signals:
- Q+1
- Q-1
- Q<<1
- Q>>1

Diagram showing the relationship between control signals and next state (Q).
An Accumulator

Values to be added are placed on A input, one per cycle. Register accumulates their sum.

Version A

This one loads 0 when CLR='1'

Version B

This one loads A when CLR='1'

Both work, they just have different timings...
Register Files

Small memories holding multiple words of data
Typical Register File

DataIn

clk
regWE

Addr

RegFile

DataOut

n

m

n

n
Typical Register File

Data to be written to register file

DataIn

 clk

 regWE

 Addr

 RegFile

 DataOut
Typical Register File

Data to be written to register file

Data that is read from register file
Typical Register File

Data to be written to register file

Data that is read from register file

Address that reads and writes are for

ECE238L
Typical Register File

Data to be written to register file

Data that is read from register file

This register will hold $2^m$ words, each $n$ bits wide

Address that reads and writes are for
Typical Register File

Data to be written to register file

Data that is read from register file

This register will hold $2^m$ words, each $n$ bits wide

Address that reads and writes are for

Controls whether reading or writing

DataIn

RegFile

DataOut

$n$

 clk

 addr

 regWE

$n$

$m$
Typical Register File

Reads are *asynchronous* (combinational)

Writes occur on the clock edge.

Controls whether reading or writing

Address that reads and writes are for

This register will hold $2^m$ words, each $n$ bits wide

Data to be written to register file

Data that is read from register file

DataIn $\rightarrow$ RegFile $\rightarrow$ DataOut

$\downarrow n$

$\rightarrow \text{clk}$

$\rightarrow \text{regWE}$

$\downarrow \text{Addr}_m$

$\downarrow n$
Building a Register File

Write Decoder

Register write signals

Reg0
Reg1
Reg2
Reg3
Reg4
Reg5
Reg6
Reg7

8:1 MUX

DataOut

DataIn
clk

RegWE

Addr

�=3

n
Building a Register File

Asynchronous read: just a MUX

Write Decoder

Reg0
Reg1
Reg2
Reg3
Reg4
Reg5
Reg6
Reg7

Register write signals

8:1 MUX

DataOut

m=3
regWE
DataIn
clk

Addr

ECE238L

16 REGISTERS
Page 33
© 2006
Building a Register File

Loadable registers

Asynchronous read: just a MUX

Write Decoder

Reg0
Reg1
Reg2
Reg3
Reg4
Reg5
Reg6
Reg7

Register write signals

8:1 MUX

DataOut

m=3

regWE
DataIn
clk

Addr

n
n
n
n
n
n
n
n

ECE238L © 2006
Building a Register File

Loadable registers

3:8 Decoder with enable

Write Decoder

Register write signals

Reg0
Reg1
Reg2
Reg3
Reg4
Reg5
Reg6
Reg7

DataOut

8:1 MUX

m=3
regWE
DataIn
clk

Addr

Asynchronous read: just a MUX
Loadable Register

Reg0  Reg1  Reg2  Reg3  Reg4  Reg5  Reg6  Reg7

Reg write signals

DataIn  clk

regWE_k → Load

Reg_k  Q

Q_k  clk

DataIn
Write Decoder

Write Decoder

Addr

regWE

m=3

Register write signals

3:8 Decoder

Addr

regWE

m=3

Register write signals
Multi-Ported Register File

One write port, one read port.

Different write and read addresses

Can be reading from one location on same cycle you write to another location
Multi-Ported Register File

One write port
Two read ports

Can be reading from two locations on same cycle you write to another location

Useful for microprocessor design

Write Decoder

DataOut1

DataOut2

Reg0
Reg1
Reg2
Reg3
Reg4
Reg5
Reg6
Reg7

8:1 MUX

WAddr WE

DataIn

clk

RAddr1
RAddr2

Register write signals
Memories vs. Register Files

- Random Access Memory (RAM) is similar to a register file
  - Stores many multi-bit words for reading/writing
- RAM usually only single-ported
- RAM usually much, much larger
  - Mbytes instead of bytes
- RAM implementation conceptually the same as register file
  - Transistor-level implementation different due to size/usage characteristics
- RAM design beyond the scope of this class