Embedded Systems 1 (ENCE361)

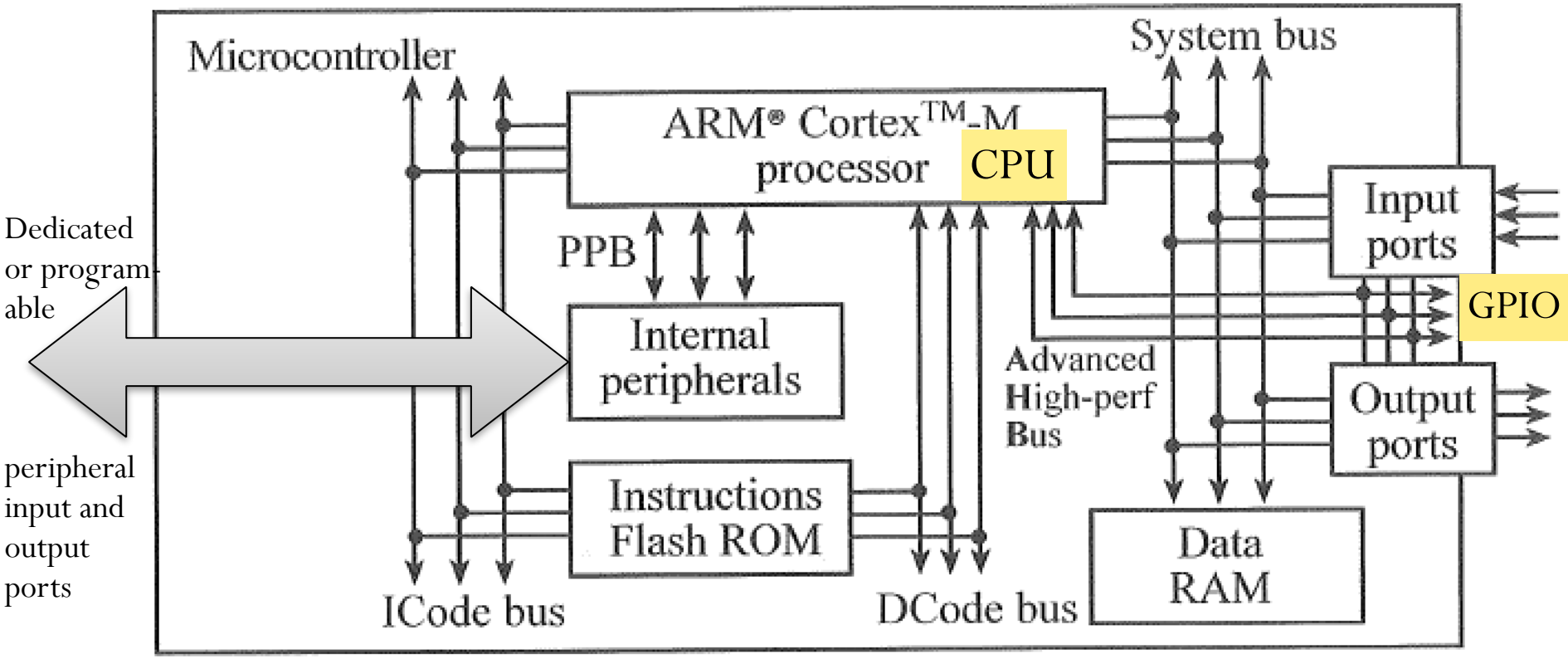
Arithmetic Logic Units (ALUs) Barrel Shifters & Register Files with ARM CPU examples
Lecture #30

By: Dr. Steve Weddell

This lecture covers the following topics:

- Some definitions
- Computer Architecture Introduction & Overview
- The Datapath - ALU, Barrel shifter, Data Buses
- The Register File
- Reference.

Overview of the Cortex-M4 MCU



From: Valvano, "Introduction to the ARM Cortex-M Microcontrollers", 2017 [4]. Annotations by Steve Weddell.

Overview of a CPU

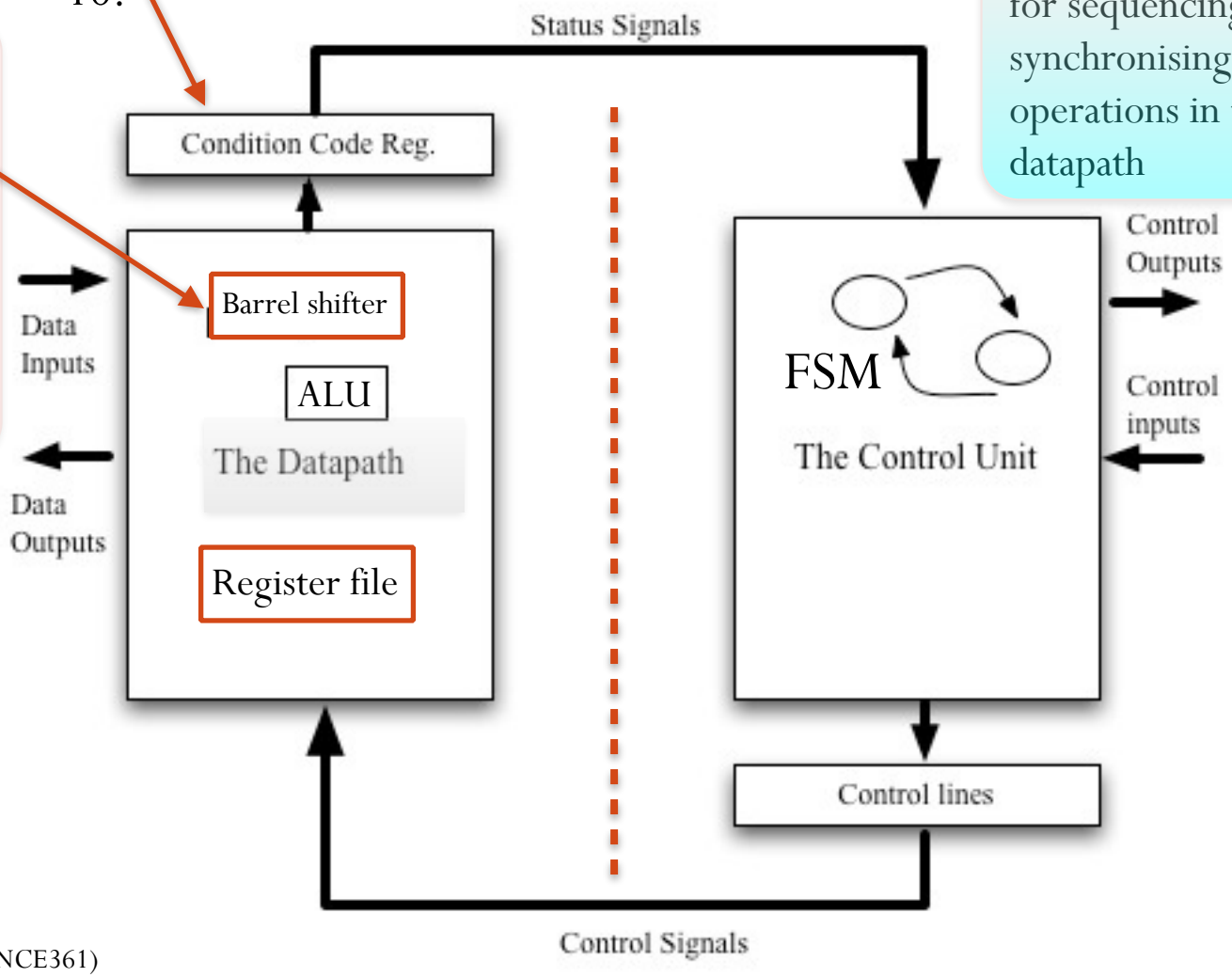
The **Datapath** comprises:

- Arithmetic & Logic Unit (ALU).
- Register File.
- Barrel shifter.
- Associated interface circuitry.

Supported condition code registers, such as *N*, *V*, *C*, and *Z*, are highlighted on Slide 10.

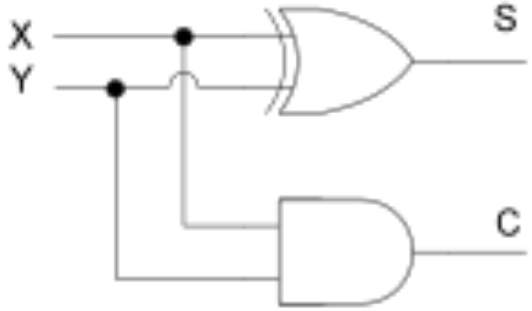
The **Control Unit** comprises:

Control circuits for sequencing/ synchronising operations in the datapath



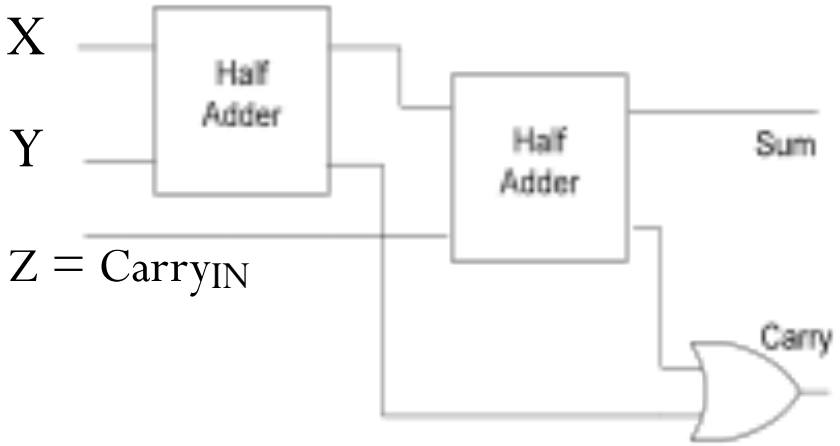
Adders (can also be used for subtraction)

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



Half-adder (no provision for previous carry)

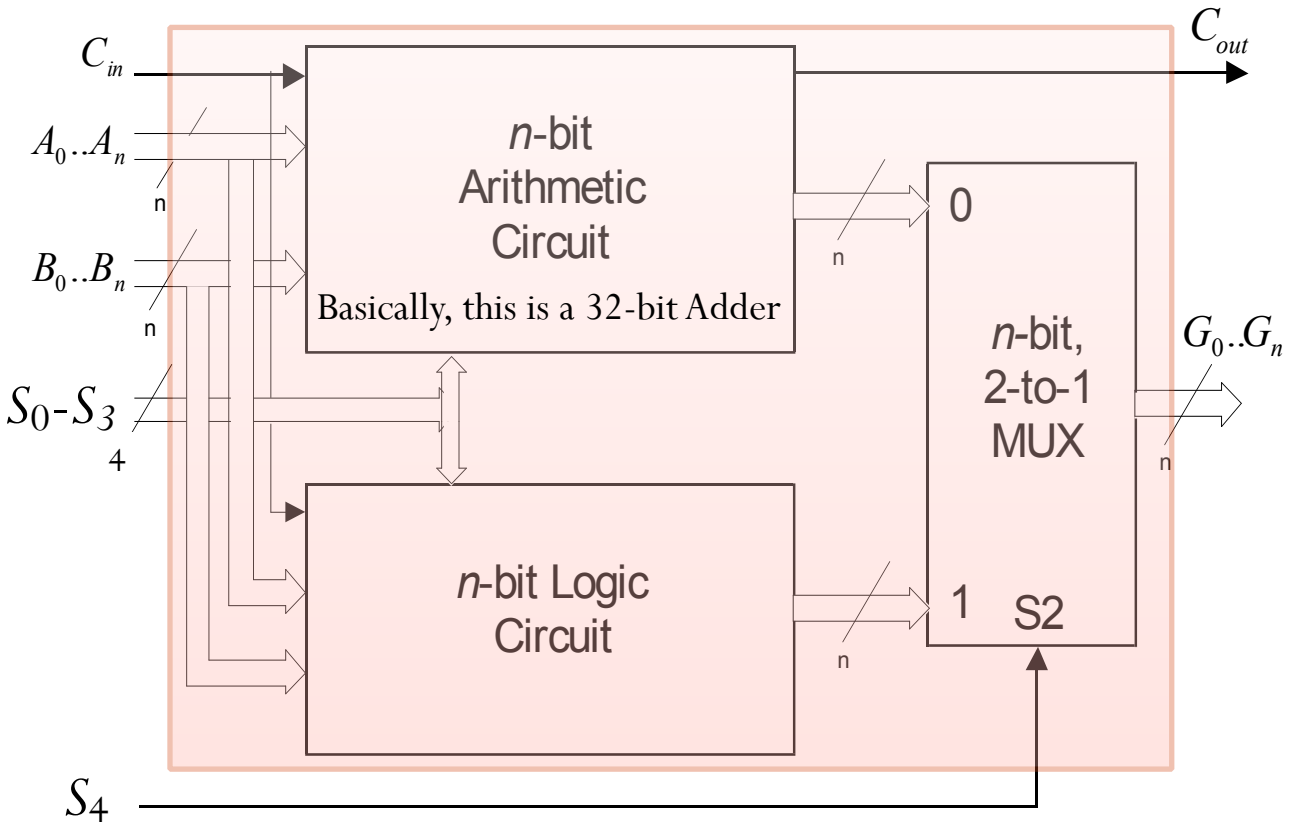
X	Y	Z	Carry	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



Q: How do you turn an adder into a subtractor?

A: You don't, you just change one of the operands

The Arithmetic & Logic Unit (ALU)



- By including an additional selection control lines, the arithmetic and logic circuitry can be combined using a multiplexer (MUX) to produce **a combined Arithmetic AND Logic Unit** (ALU) as shown.
- The following function table from Furber [1], lists ALU functions for the ARM2.

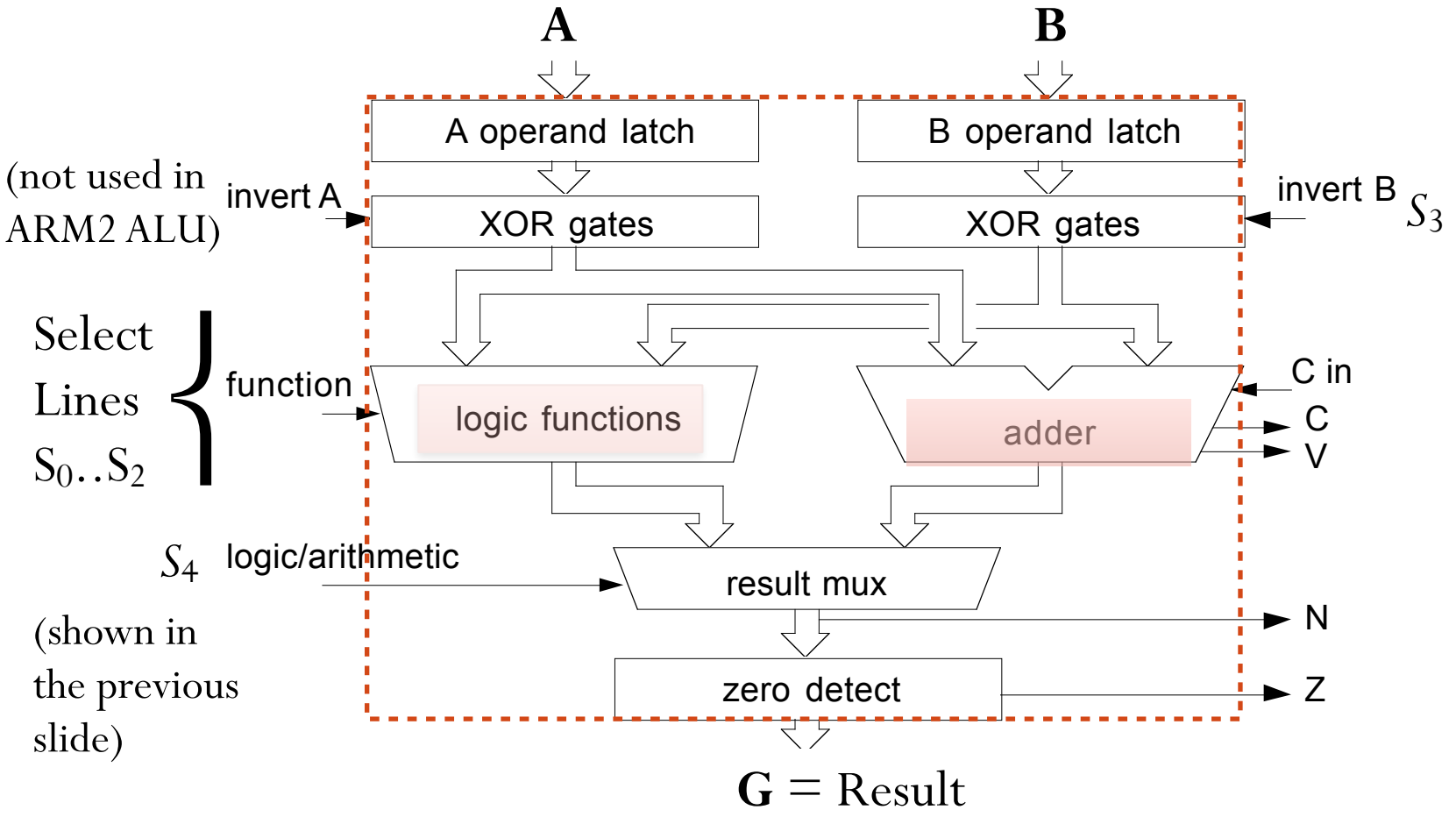
ARM-2 ALU Functions

S_5	S_4	S_3	S_2	S_1	S_0	**ALU output** (G)	
0	0	0	1	0	0	A and B	
0	0	1	0	0	0	A and not B	
0	0	1	0	0	1	A xor B	
0	1	1	0	0	1	A plus not B plus carry	* A - B
0	1	0	1	1	0	A plus B plus carry	A + B + carry
1	1	0	1	1	0	not A plus B plus carry	* B - A with borrow
0	0	0	0	0	0	A	
0	0	0	0	0	1	A or B	
0	0	0	1	0	1	B	
0	0	1	0	1	0	not B	
0	0	1	1	0	0	zero	

* Two's complement is used to implement subtractions. In the above case, a set "carry" bit is employed to convert one's compliment to two's compliment. Separate inputs, such as "Invert A" or "Invert B" (see next slide) are used to complement A or B by setting the respective input to logic-1 using XOR gates.

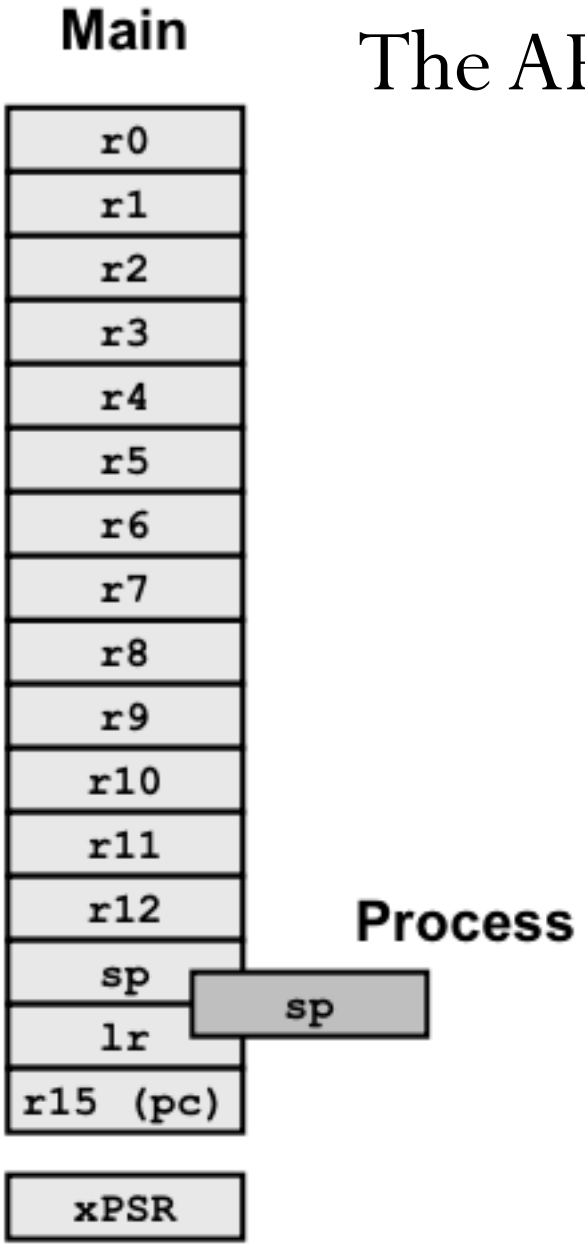
Note: logical AND can also represented with the symbol ^, the OR with the symbol ˅, and the complement (NOT) function using a bar symbol over a variable, e.g., $\bar{A}$ or $\neg A$.

The Arithmetic Logic Unit (ALU) - ARM-6



- The ALU is part of the datapath and is responsible for the execution of arithmetic and logical operations.
- **Exercise**: If A = 7, B = 3 and Cin = 1, carry out a two's compliment subtraction of A - B by hand.

The ARM Cortex-M4 register file*



The register file provides source and destination operands for the ALU

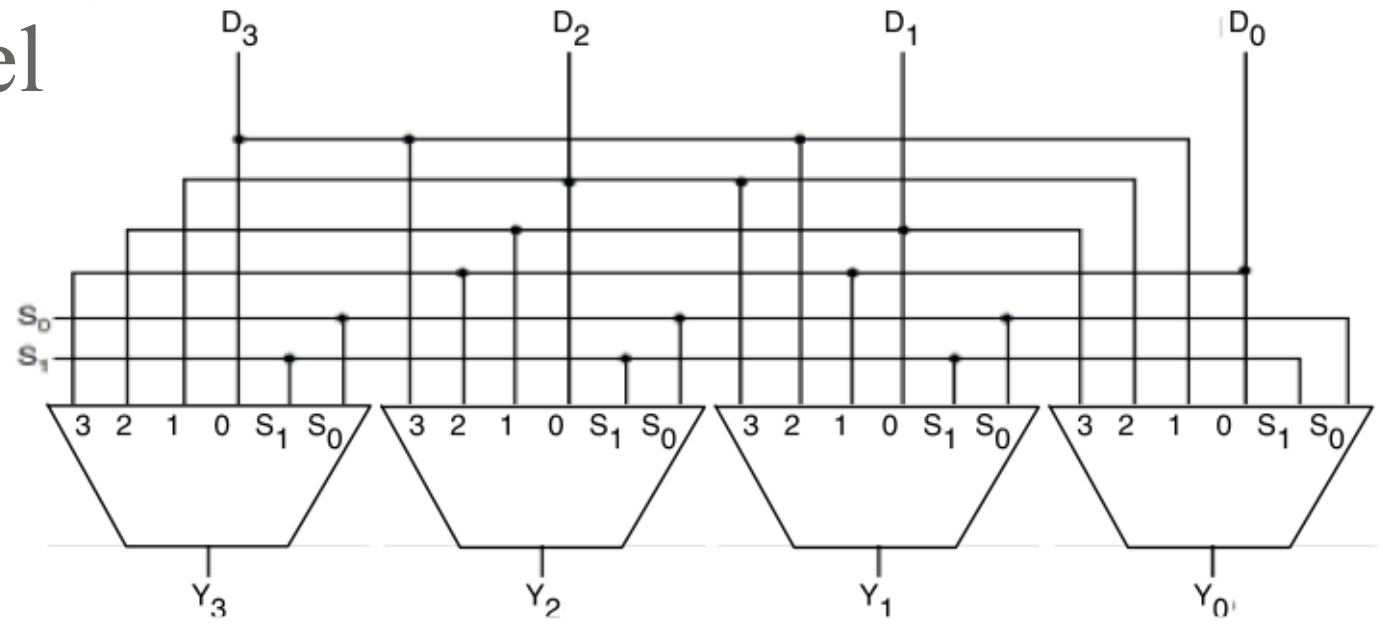
- Simpler than the ARM7 TDMI, i.e., no banked registers (except for SP).
- All resisters are 32-bits.
- sp: stack pointer
- lr: link register
- pc: program counter.

*From an ARM presentation

Bit shifting using a *Barrel Shifter*

- A *shift module* allows a group of data bits within a register to be shifted (as a group), either one place to the left or right.
- A *barrel shifter*, allows left or right shifts, by one or more bits, without iteration.
- A barrel shifter module is typically part of the Datapath within a CPU, as shown on Slide 7.
- Bi-directional shifts are possible due to the rotational characteristic of barrel shifters. For example, given a 4-bit barrel shifter, shift right by one is similar to shift left by three.
- A 4-bit example [3] is shown on the following page. Another configuration is given by Furber [1].

A barrel shifter



Function Table for 4-Bit Barrel Shifter

Select		Output				
S_1	S_0	Y_3	Y_2	Y_1	Y_0	Operation
0	0	D_3	D_2	D_1	D_0	No rotation
0	1	D_2	D_1	D_0	D_3	Rotate one position
1	0	D_1	D_0	D_3	D_2	Rotate two positions
1	1	D_0	D_3	D_2	D_1	Rotate three positions

Note: S_0 & S_1 select the # of left shifts

References

[1] Furber, S., *ARM system-on-chip architecture*, 2nd Ed., Addison-Wesley, 2000.

[2] Atmel Corporation, *AT91 ARM Thumb-based Microcontrollers Datasheet*, Preliminary, November, 2006.

[3] M.M. Mano, & C.R. Kime. *Logic and Computer Design Fundamentals*, 2nd Ed., Prentice Hall, 2001.

[4] Valvano, "Introduction to the ARM Cortex-M Microcontrollers", 2017
Note: This book will be placed in the short-term reserve in the EPS library.

Exercises

The ARM7-TDMI has a banked register file, however the Cortex M series does not support this. How are fast interrupts and other CPU conditions supported without bank register switching?

If complex Load and Store operations take more than one instruction cycle to complete, how are low-latency interrupts maintained using the Cortex M3?

Why is it necessary that the ISR number is supported in the Program Status Register?