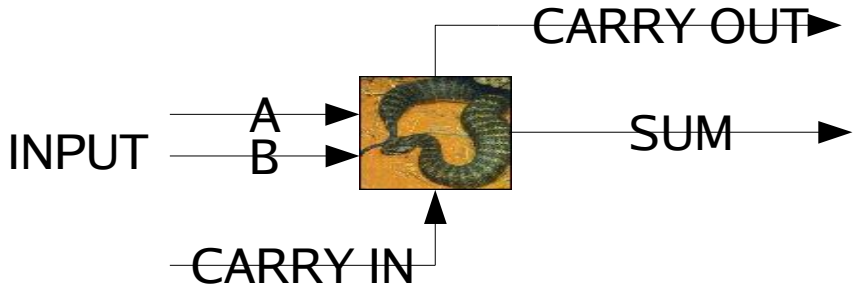
CMSC 11:
Introduction to Computer Science

Jaderick P. Pabico <jppabico@uplb.edu.ph>
Institute of Computer Science, CAS, UPLB

Review

- Translate/convert binary numbers to decimal
- Binary arithmetic
  - Addition: 5 rules
  - Subtraction: addition using the two's complement
  - Multiplication/division: repeated addition/subtraction
- Terminologies – bit, munch, nibble, byte, word
- The ADDER – 4-bit adder
  - 4 1-bit adders

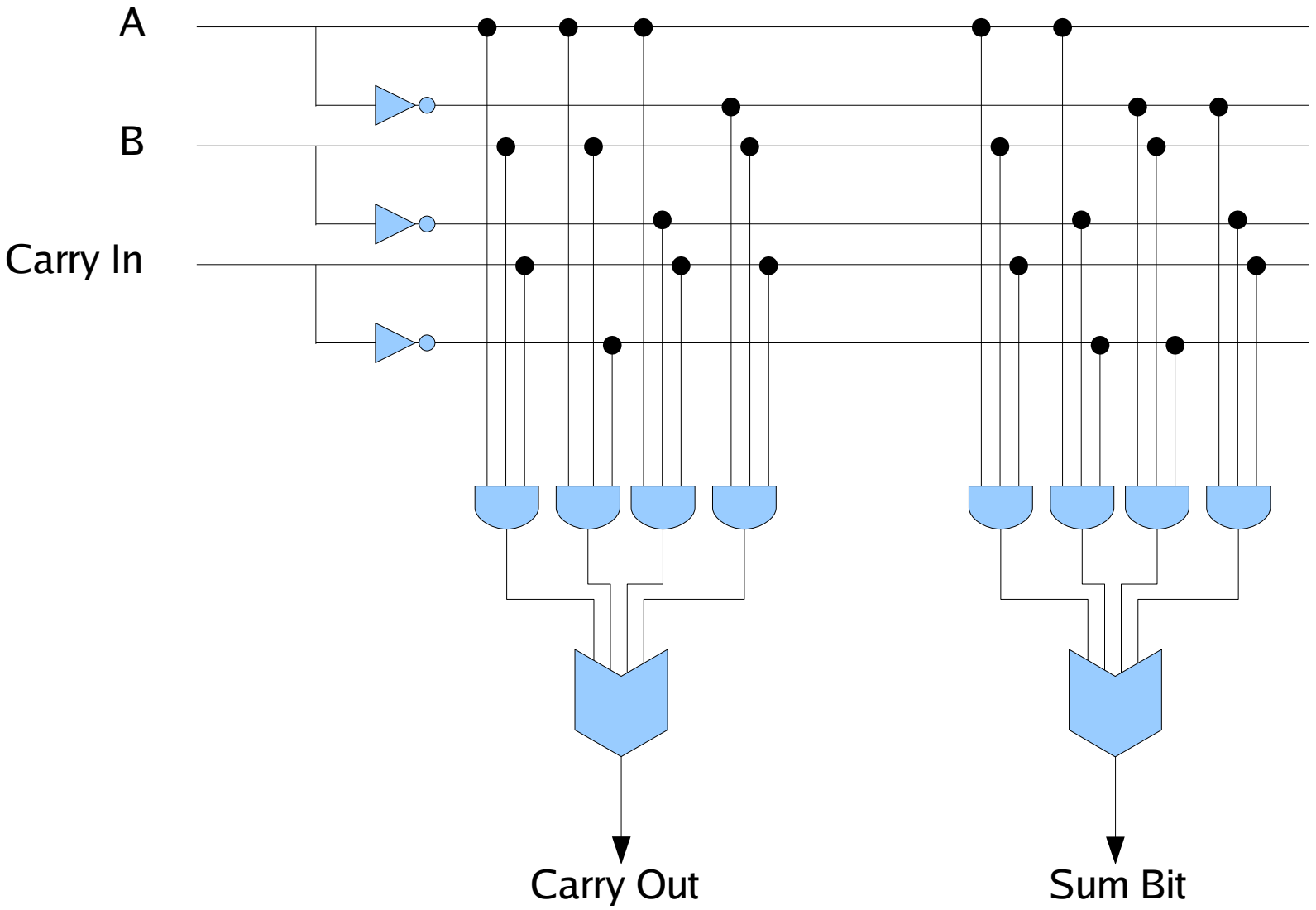


The I/O Table of the 1-bit adder

- From last meeting's last slide:

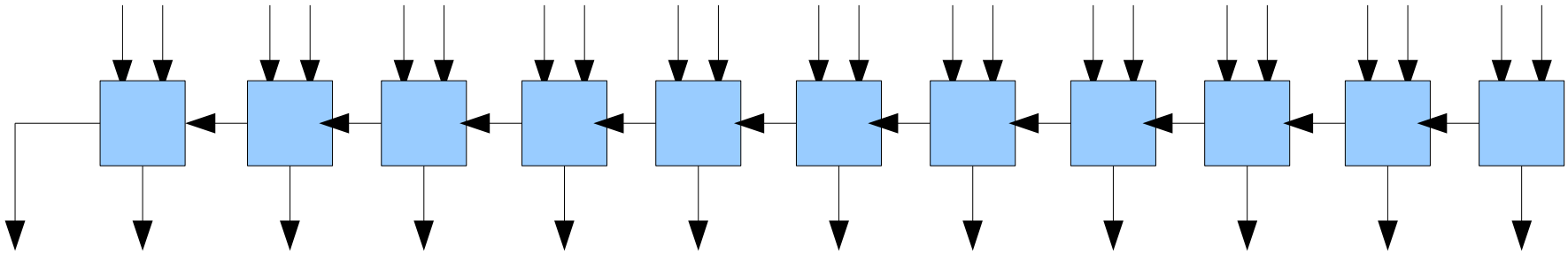
A	B	Carry In	Carry Out	SUM BIT
1	1	1	1	1
1	1	0	1	0
1	0	1	1	0
1	0	0	0	1
0	1	1	1	0
0	1	0	0	1
0	0	1	0	1
0	0	0	0	0

Logic diagram of 1-bit adder



A sequence of 1-bit adders

- You can add two numbers of any length by hooking together enough 1-bit adders



Implication

- Binary is the “natural” system for encoding numbers in a machine made of on-off switches
- However, computers use several variations on the basic idea.
- INTEGERS
  - or whole numbers
  - encoded in straight binary (if not too large)
  - example: 185
    - would become 10111001

Implication

- FLOATING POINT
  - for large or fractional numbers
  - for example: 19,700,030.2
    - would be encoded as the binary equivalent of 197 and 5
      - meaning 197 x 10^5
      - floating point representation often involves rounding off
- BINARY CODED DECIMAL
  - represents a number in decimal
  - each digit is encoded in binary
  - Example: 967 = 1001 0110 0111

Implication

- What about non-numerical information
  - alphabet
  - punctuation marks
  - other symbols, even blank space?
- No natural way to encode these into 0s and 1s
- Computer scientists invented and adopted a standard code by mutual agreement
  - ASCII

Actually, ASCII is used by everyone but IBM, which has its own code, called EBCDIC

Unity! Where is unity?

ASCII

- The American Standard Code for Information Interchange

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	“	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

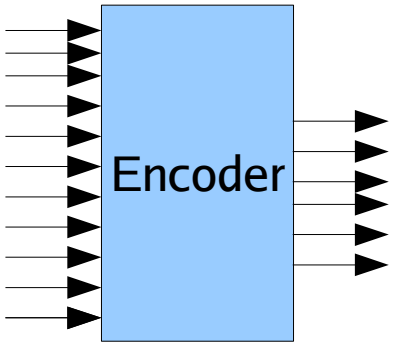
ASCII

- Thus, the letter “T” is encoded as 101 0100
- Ths first two columns contain symbols for such things as “start of heading” (SOH) and other textual directions

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	“	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

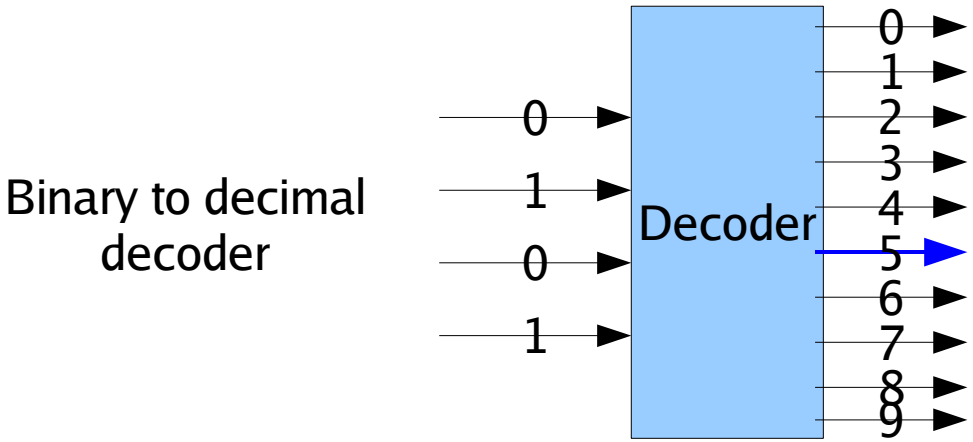
Encoders/Decoders

- To encode and decode data, computers use logic devices called, naturally enough, ENCODERS and DECODERS
- An ENCODER
  - has many inputs
  - few outputs
  - A single input signal produces a pattern of outputs
  - For example, a computer keyboard is attached to an encoder which translates a single keystroke into its ASCII code.



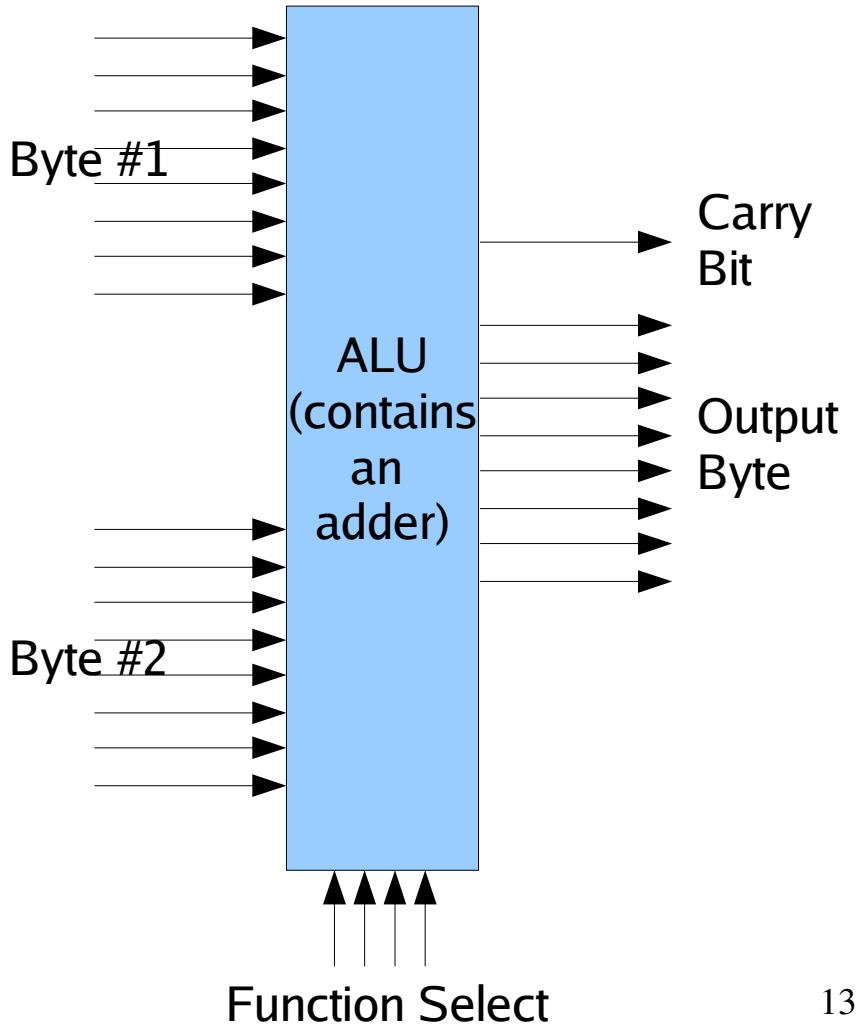
Encoders/Decoders

- A DECODER
  - works the other way around
  - For example
    - One that converts a binary nibble into a decimal digit
    - Another transforms a specified location (address) in memory into a signal



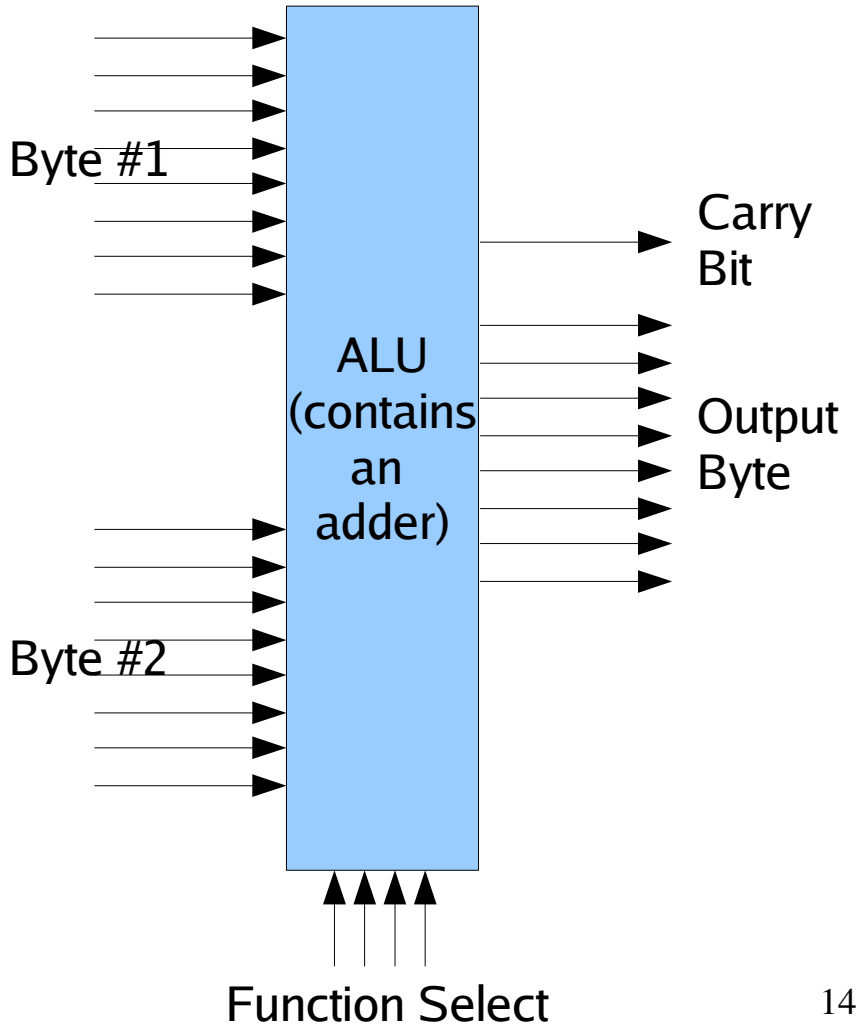
Arithmetic Logic Unit

- Once alphanumeric information is encoded in binary strings, it is ready to be processed by the computer's most elaborate combination of logic gates
- THE ARITHMETIC LOGIC UNIT
- ALU for short



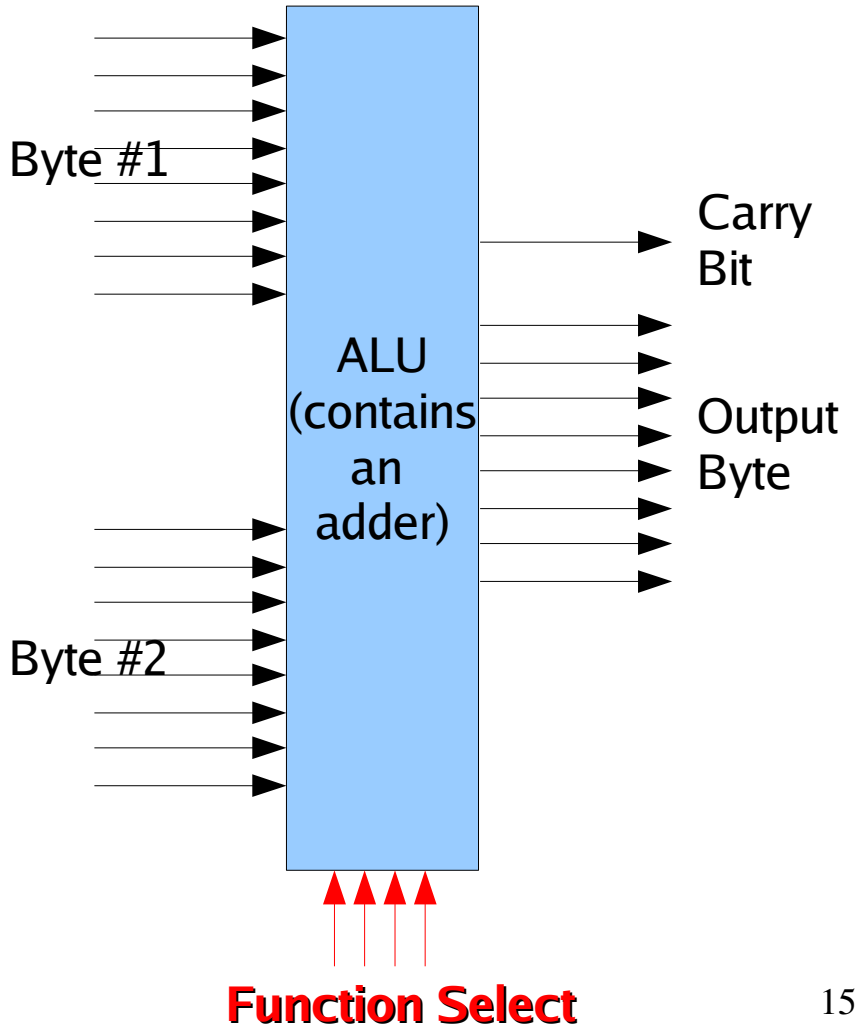
Arithmetic Logic Unit

- ALU is the machine's central processor
- It can
  - add, subtract, multiply
  - compare, shift
  - and perform a wealth of other logical operations
- Example: 8-bit ALU
- Can range from 4 to 64



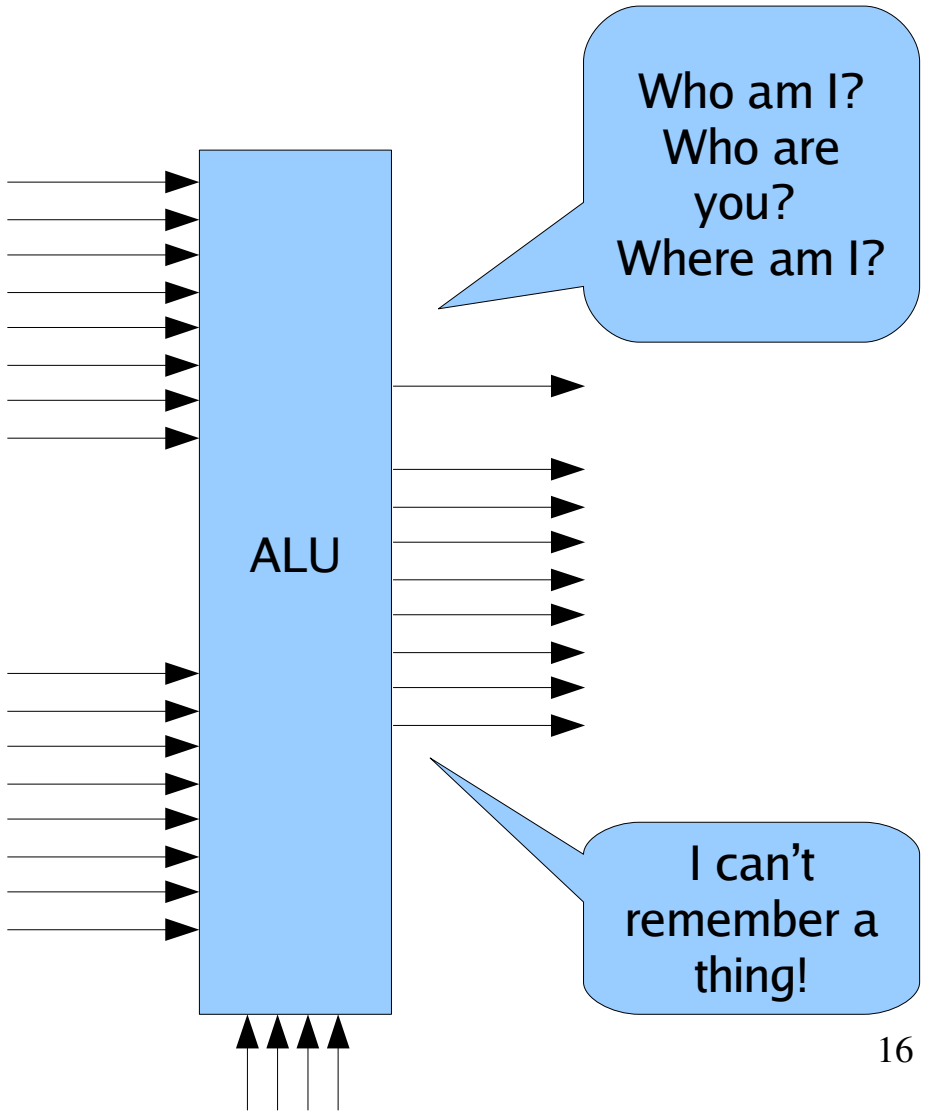
Arithmetic Logic Unit

- The FUNCTION SELECT inputs
  - determine which arithmetic or logical function the ALU is to perform
  - each function having its own binary code
  - example: 0001 would mean ADD



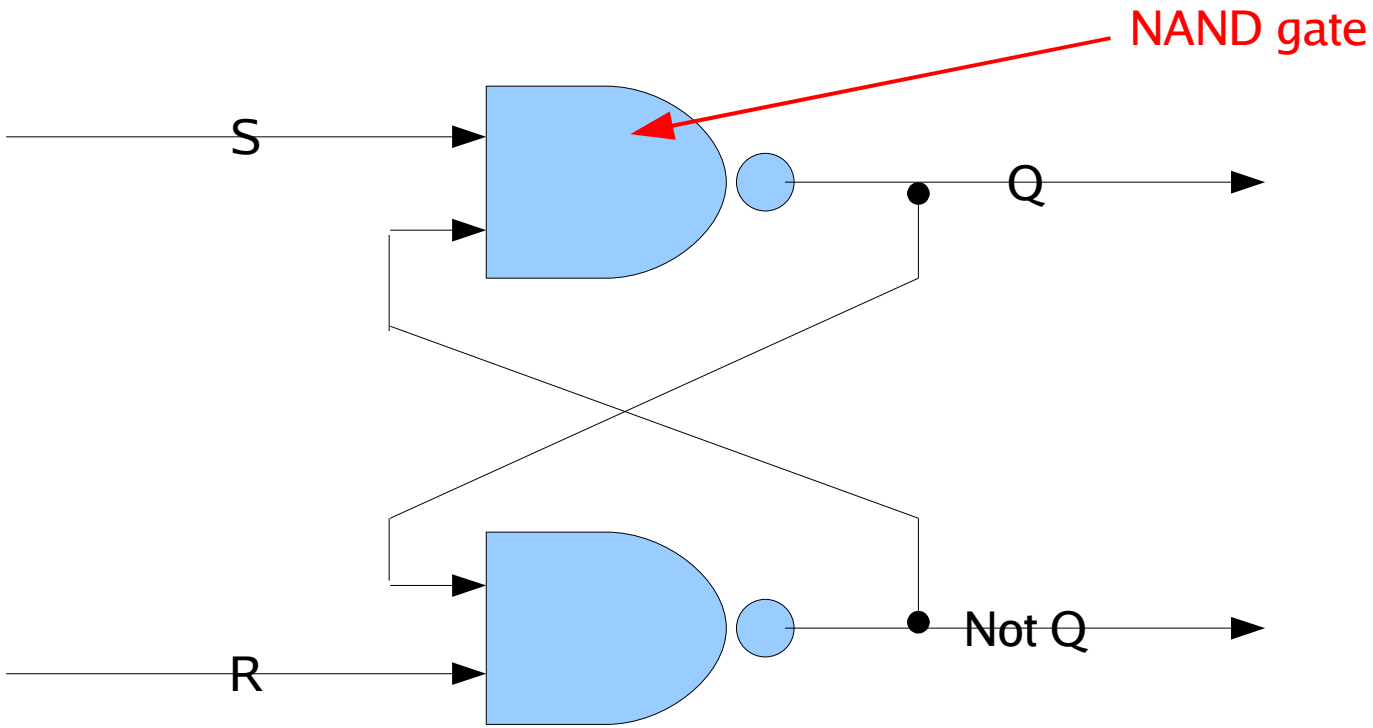
Arithmetic Logic Unit

- The ALU would be a complete central processing unit except for one thing:
- It is unable to STORE results!
- Although the ALU can perform miracles, it can't REMEMBER anything



Flip-Flop

- This is where FLIP-FLOPs come in



NAND-gate

- Short for Not AND
- Its I/O Table is:

1	1	0
1	0	1
0	1	1
0	0	1

- And same as: