

# Processors, FPGAs, and ASICs

Stephen A. Edwards

Columbia University

Spring 2019

# Spectrum of IC choices

**Flexible, efficient**

**You choose**



Full Custom

Polygons (Intel)

ASIC

Circuit (Sony)

Gate Array

Wires

FPGA

Logic network

PLD

Logic function

GP Processor

Program (e.g., ARM)

SP Processor

Program (e.g., DSP)

Multifunction

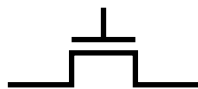
Settings (e.g., Ethernet Ctrl.)

Fixed-function

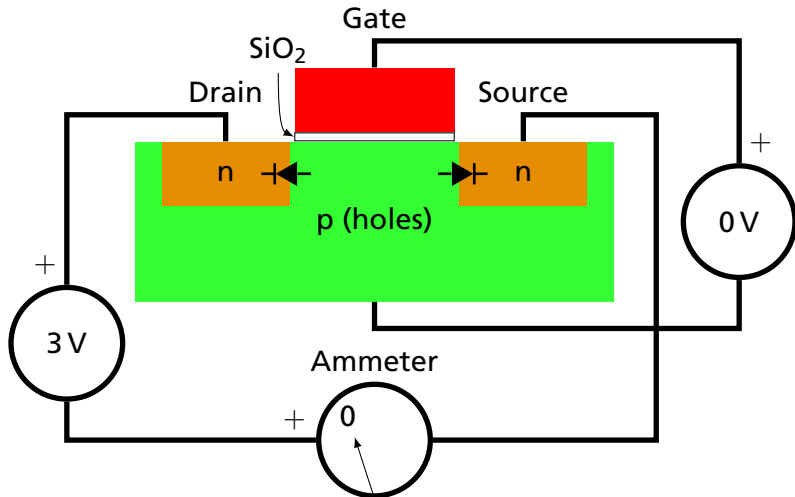
Part number (e.g., 74HCT00)

**Cheap, quick to design**

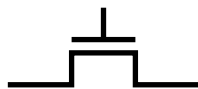
# An N-Channel MOS Transistor



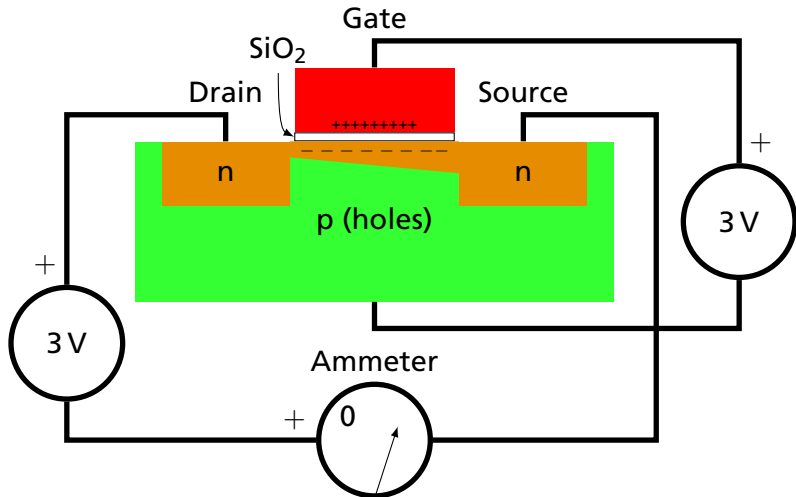
Gate at 0V: Off



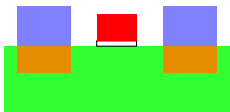
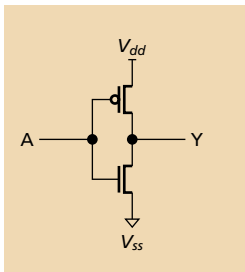
# An N-Channel MOS Transistor



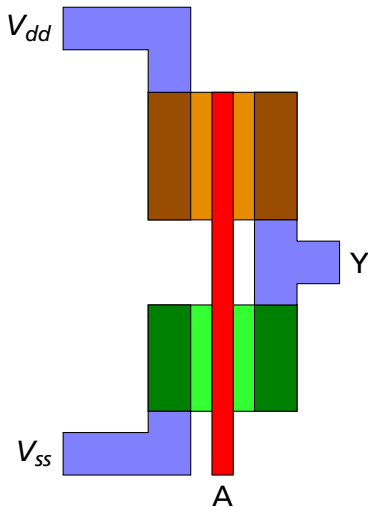
Gate positive: On



# CMOS Inverter Layout

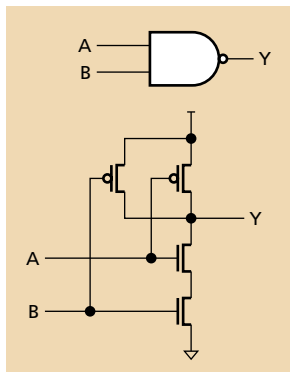


Cross Section Through  
N-channel FET



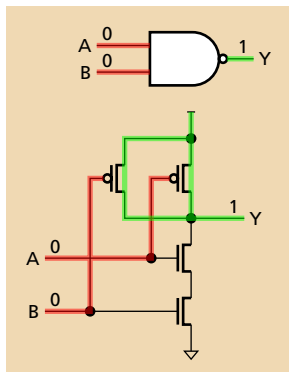
Top View

# The CMOS NAND Gate



Two-input NAND gate:  
two n-FETs in series;  
two p-FETs in parallel

# The CMOS NAND Gate

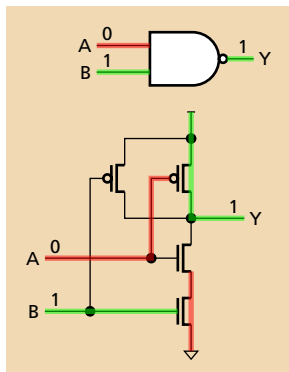


Both inputs 0:

Both p-FETs turned on

Output pulled high

# The CMOS NAND Gate



One input 1, the other 0:

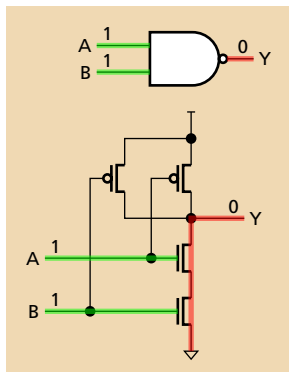
One p-FET turned on

Output pulled high

One n-FET turned on, but does not control output



# The CMOS NAND Gate



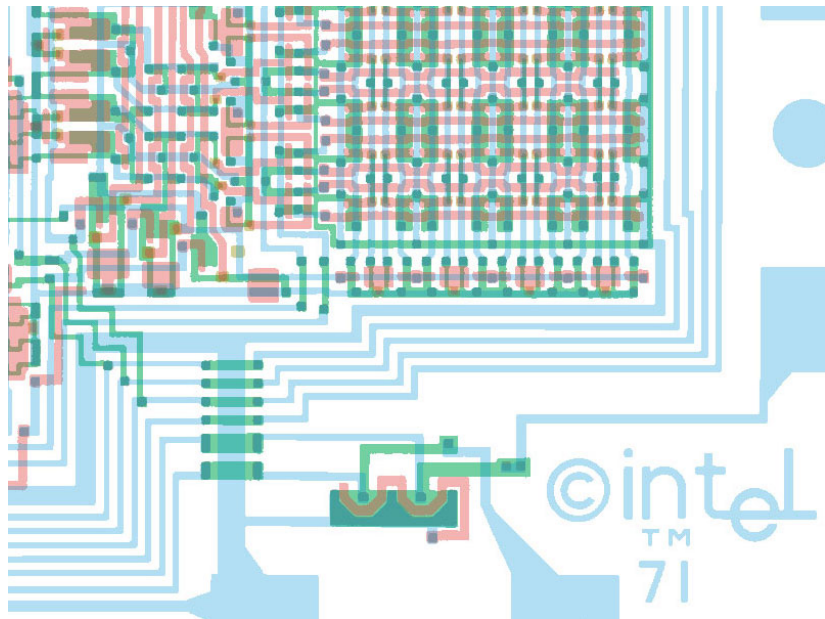
Both inputs 1:

Both n-FETs turned on

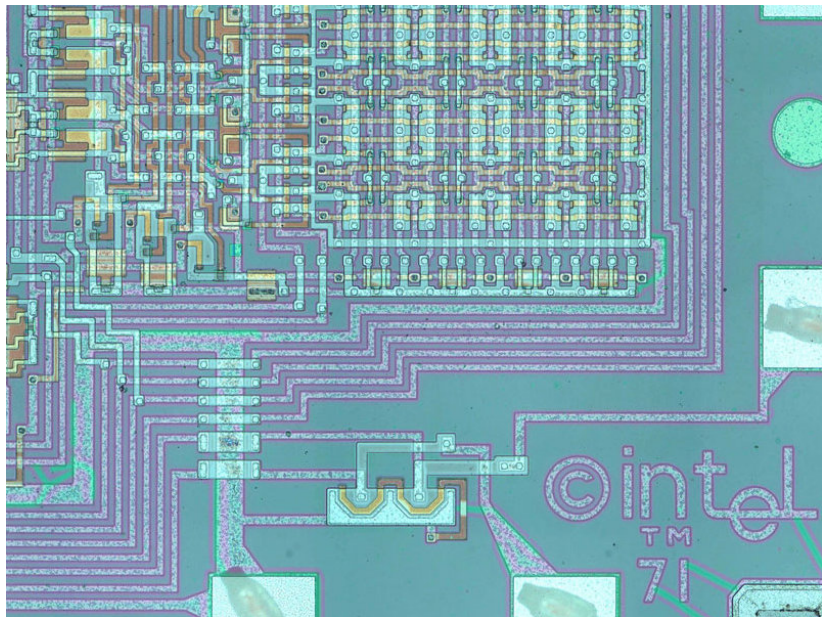
Output pulled low

Both p-FETs turned off

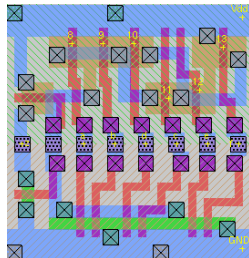
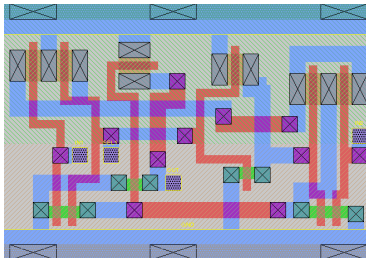
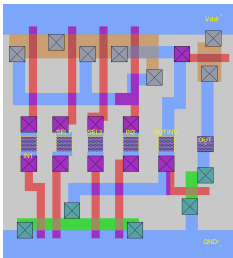
## Full Custom: Intel 4004 Masks (23K Transistors)



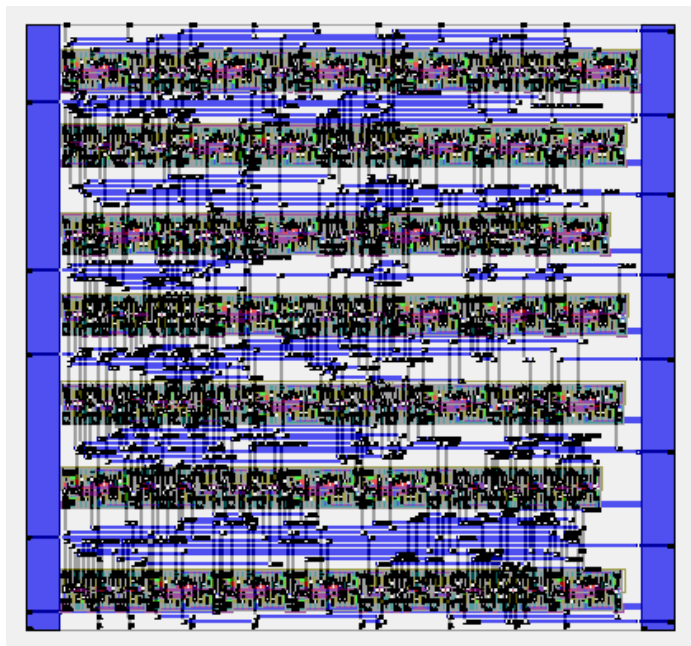
# Full Custom: Intel 4004 Die Photograph



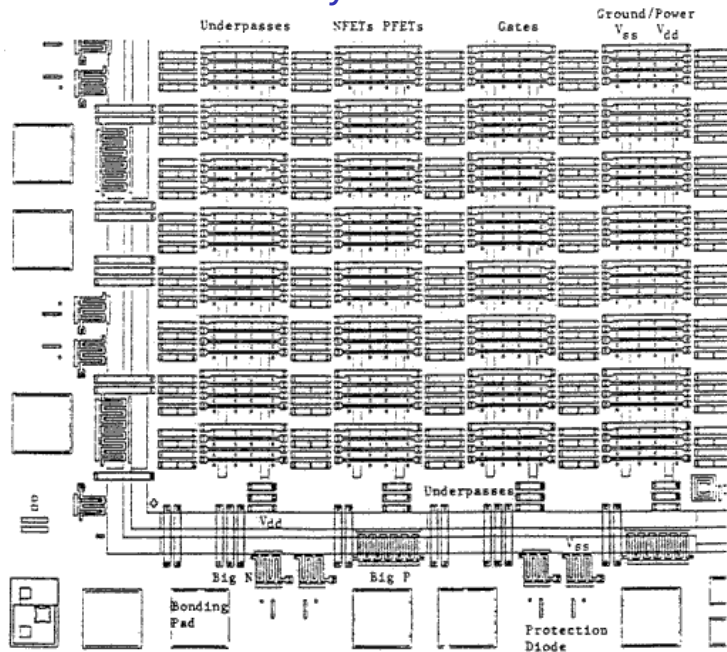
# Standard Cell ASICs



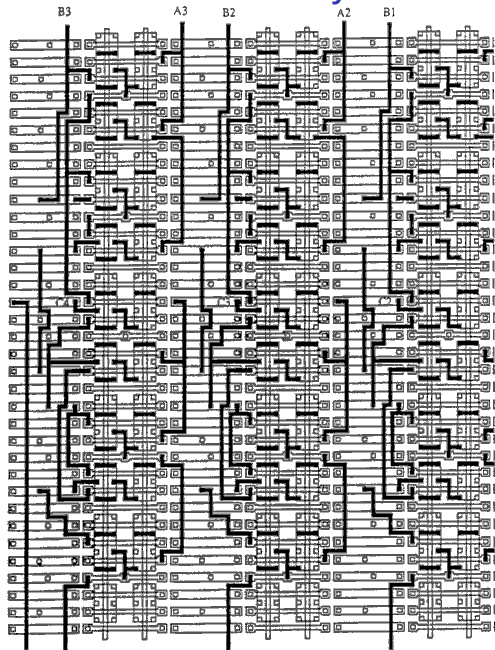
## Standard Cell ASICs



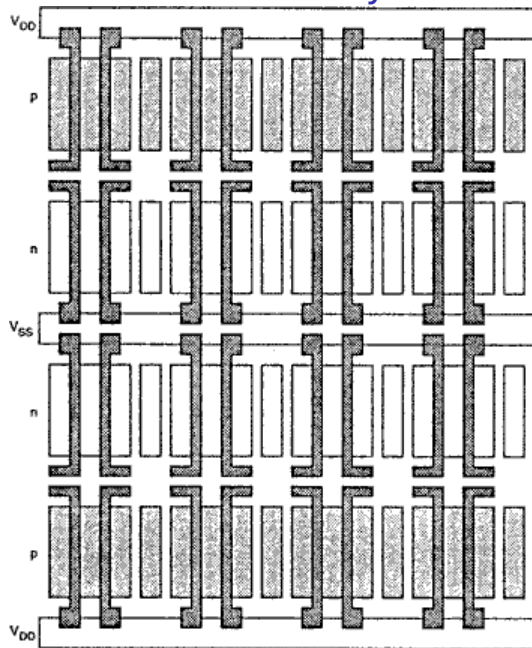
# Channeled Gate Arrays



# Channeled Gate Arrays

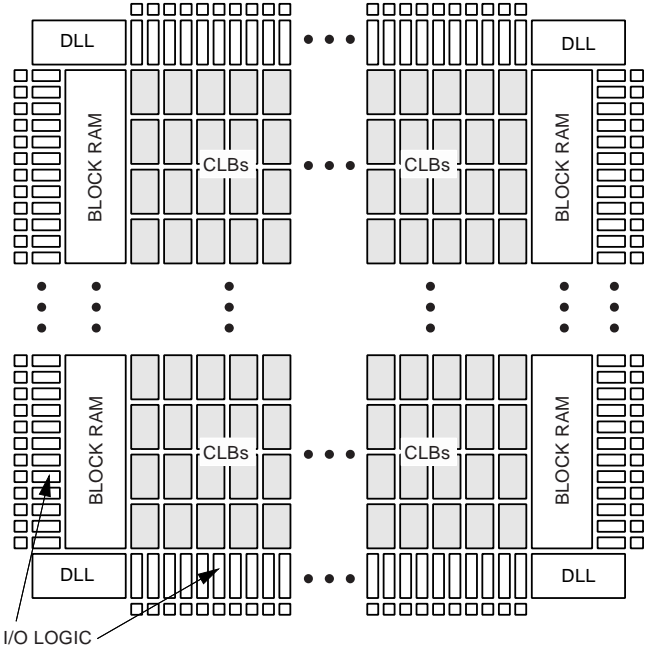


## Sea-of-Gates Gate Arrays

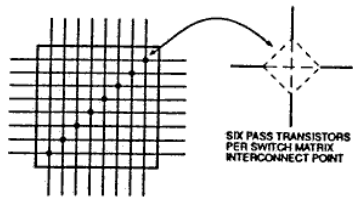




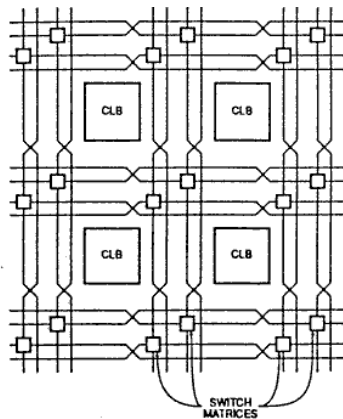
# FPGAs: Floorplan



# FPGAs: Routing

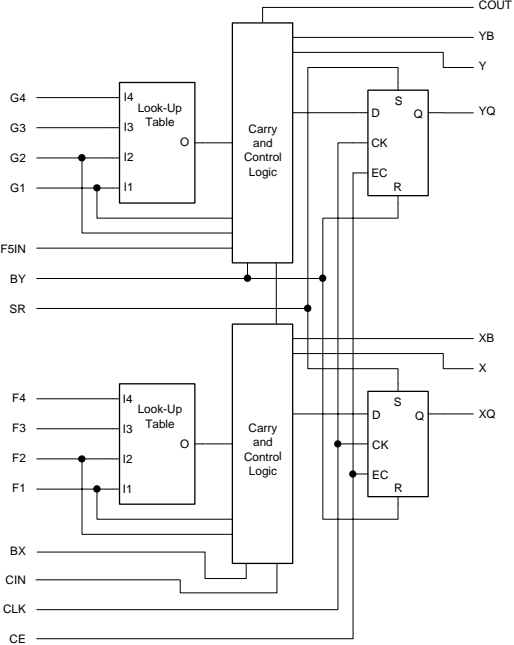


Single-length line Switch Matrix connections

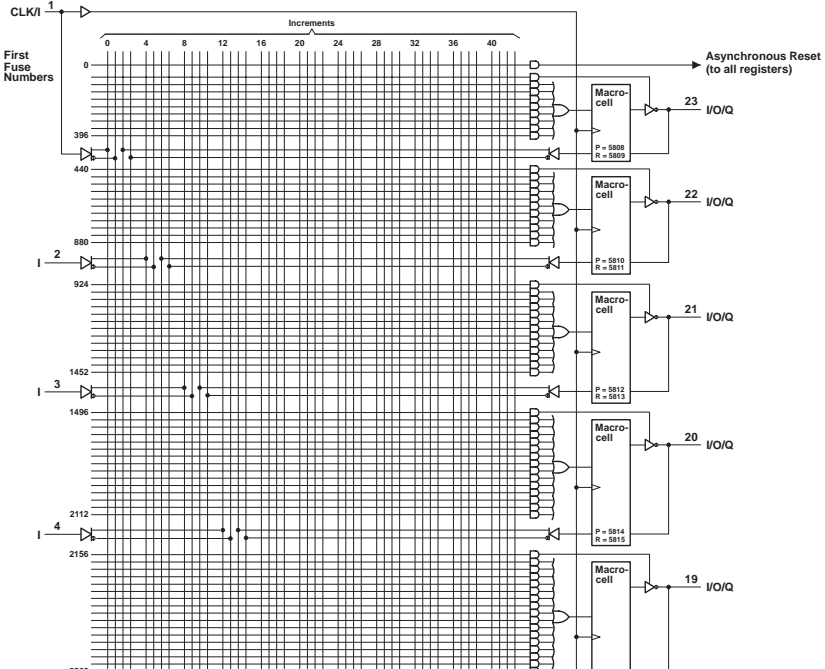


Double-length lines in CLB array

# FPGAs: CLB



# PLAs/CPLDs: The 22v10



## Example: Euclid's Algorithm

```
int gcd(int m, int n)
{
    int r;
    while ((r = m % n) != 0) {
        m = n;
        n = r;
    }
    return n;
}
```

# i386 Programmer's Model

31	0	
eax		Mostly
ebx		General-
ecx		Purpose
edx		Registers
esi		Source index
edi		Destination index
ebp		Base pointer
esp		Stack pointer
eflags		Status word
eip		Instruction Pointer

15	0	
cs		Code segment
ds		Data segment
ss		Stack segment
es		Extra segment
fs		Data segment
gs		Data segment

## Euclid on the i386

```
gcd:  pushl  %ebp
      movl  %esp,%ebp
      pushl %ebx
      movl  8(%ebp),%eax
      movl  12(%ebp),%ecx
      jmp   .L6
.L4:  movl  %ecx,%eax
      movl  %ebx,%ecx
.L6:  cltd
      idivl %ecx
      movl  %edx,%ebx
      testl %edx,%edx
      jne   .L4
      movl  %ecx,%eax
      movl  -4(%ebp),%ebx
      leave
      ret
```

# SPARC Programmer's Model

31	0			31	0	
r0		Always 0		r24/i0		Input Registers
r1		Global Registers		⋮		
⋮				r30/i6		Frame Pointer
r7				r31/i7		Return Address
r8/o0		Output Registers		PSW		Status Word
⋮				PC		Program Counter
r14/o6		Stack Pointer		nPC		Next PC
r15/o7						
r16/l0		Local Registers				
⋮						
r23/l7						

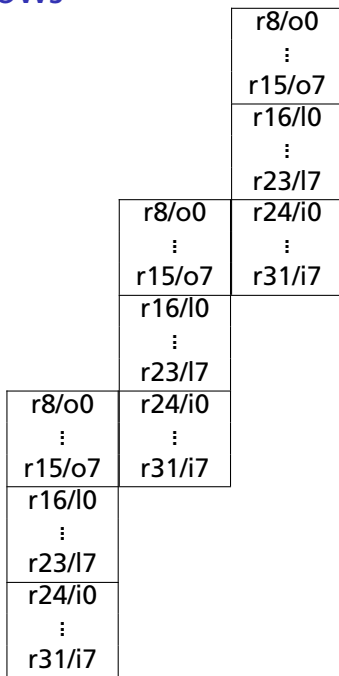


# SPARC Register Windows

The output registers of the calling procedure become the inputs to the called procedure

The global registers remain unchanged

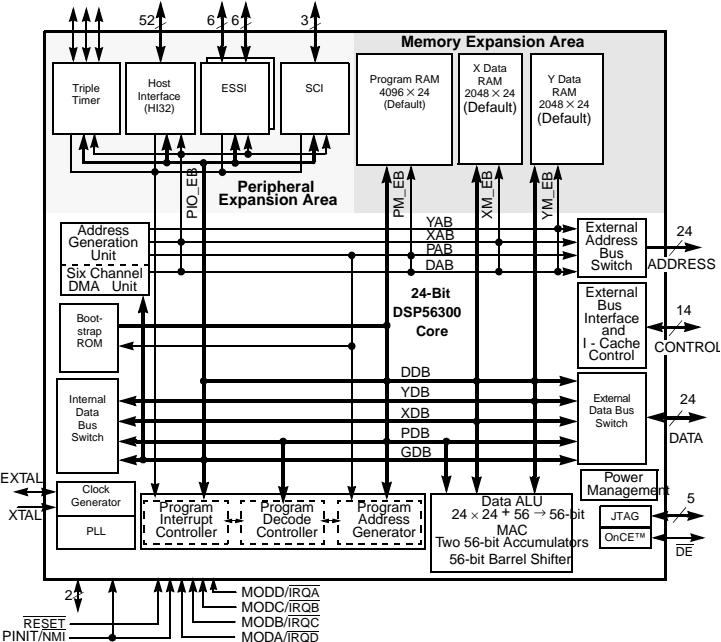
The local registers are not visible across procedures



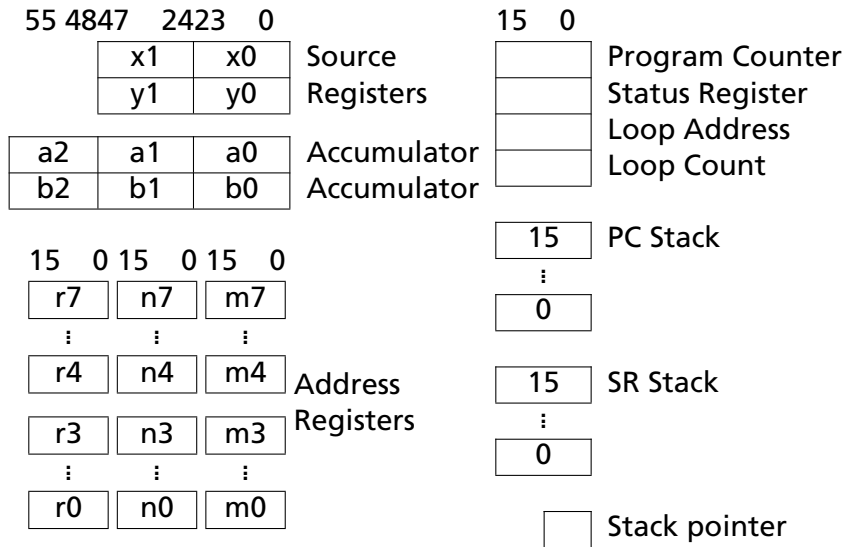
# Euclid on the SPARC

```
gcd:
    save %sp, -112, %sp
    mov %i0, %o1
    b   .LL3
    mov %i1, %i0
    mov %i0, %o1
    b   .LL3
    mov %i1, %i0
.LL5:
    mov %o0, %i0
.LL3:
    mov %o1, %o0
    call .rem, 0
    mov %i0, %o1
    cmp %o0, 0
    bne .LL5
    mov %i0, %o1
    ret
    restore
```

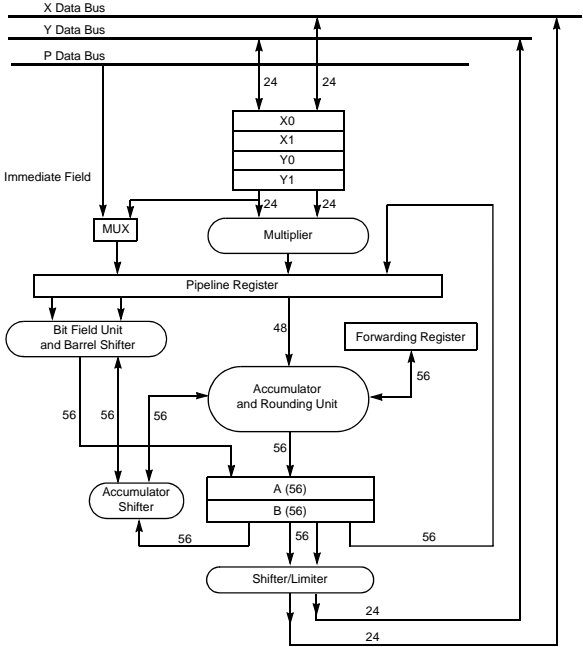
# Motorola DSP56301



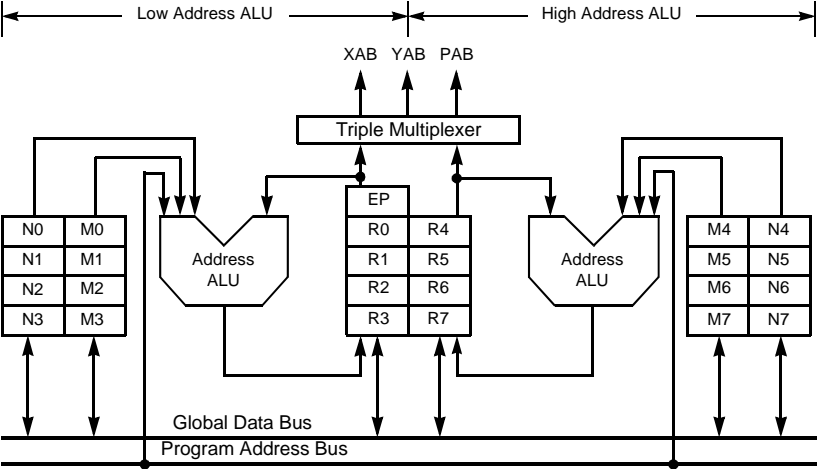
# DSP 56000 Programmer's Model



# Motorola DSP56301 ALU



# Motorola DSP56301 AGU



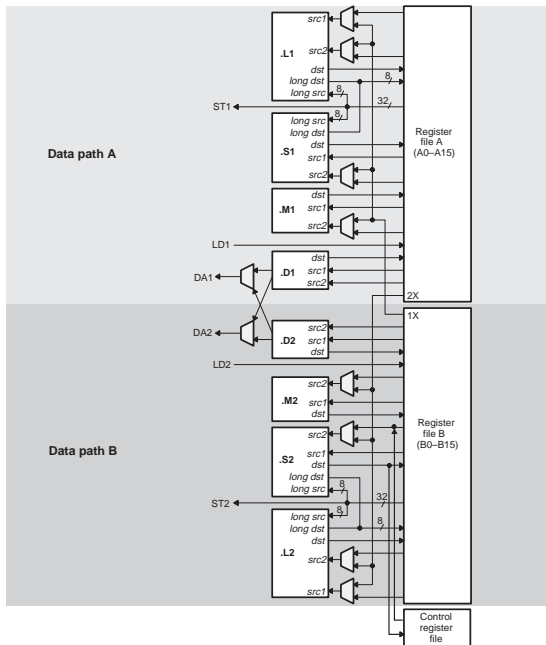
## FIR Filter in 56000

```
move    #samples, r0
move    #coeffs, r4
move    #n-1, m0
move    m0, m4
movep   y:input, x:(r0)
clr     a          x:(r0)+, x0   y:(r4)+, y0

rep     #n-1
mac     x0,y0,a    x:(r0)+, x0   y:(r4)+, y0

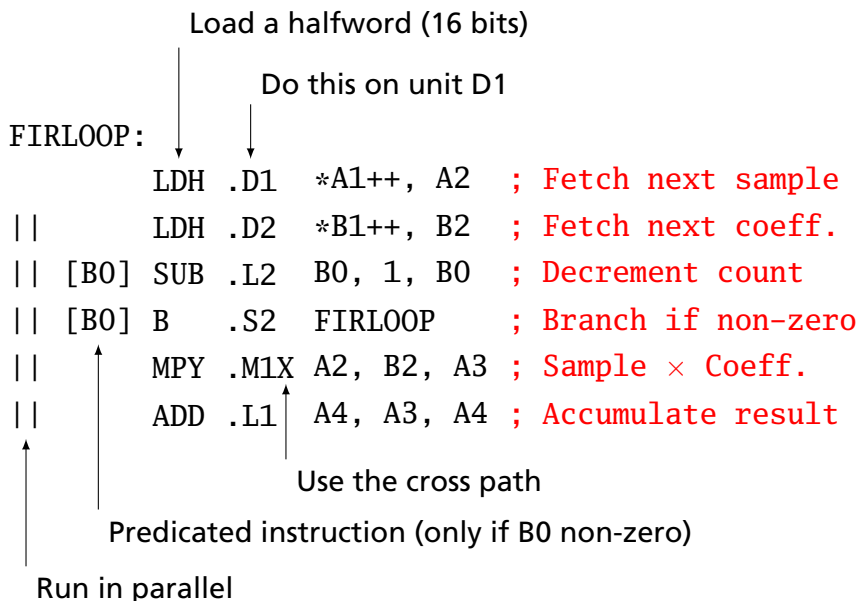
macr    x0,y0,a    (r0)-
movep   a, y:output
```

# TI TMS320C6000 VLIW DSP

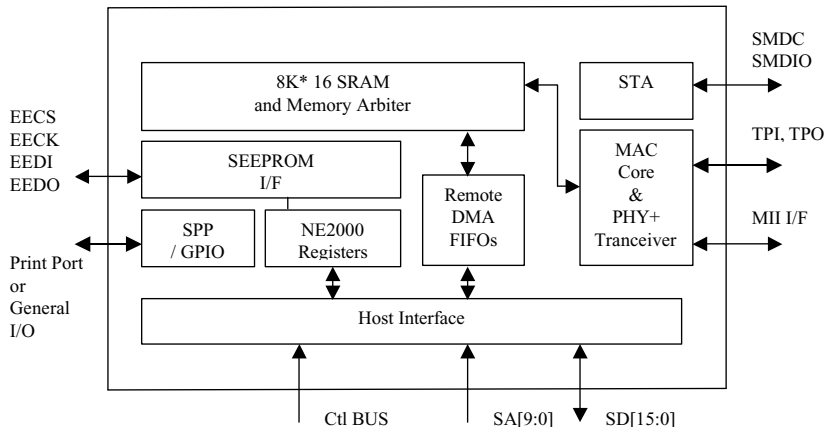




## FIR in One 'C6 Assembly Instruction



# AX88796 Ethernet Controller

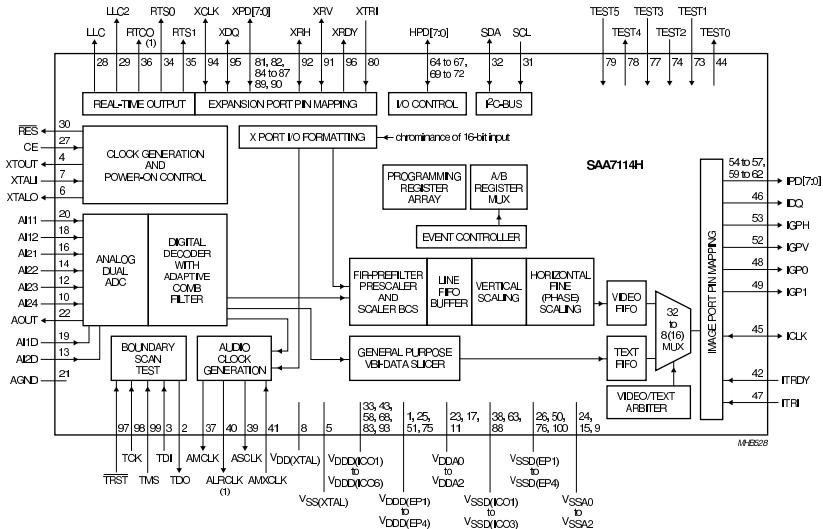


# Ethernet Controller Registers

PAGE 0 (PS1=0,PS0=0)

<b>OFFSET</b>	<b>READ</b>	<b>WRITE</b>
00H	Command Register ( CR )	Command Register ( CR )
01H	Page Start Register ( PSTART )	Page Start Register ( PSTART )
02H	Page Stop Register ( PSTOP )	Page Stop Register ( PSTOP )
03H	Boundary Pointer ( BNR )	Boundary Pointer ( BNR )
04H	Transmit Status Register ( TSR )	Transmit Page Start Address ( TPSR )
05H	Number of Collisions Register ( NCR )	Transmit Byte Count Register 0 ( TBCR0 )
06H	Current Page Register ( CPR )	Transmit Byte Count Register 1 ( TBCR1 )
07H	Interrupt Status Register ( ISR )	Interrupt Status Register ( ISR )
08H	Current Remote DMA Address 0 ( CRDA0 )	Remote Start Address Register 0 ( RSAR0 )
09H	Current Remote DMA Address 1 ( CRDA1 )	Remote Start Address Register 1 ( RSAR1 )
0AH	Reserved	Remote Byte Count 0 ( RBCR0 )
0BH	Reserved	Remote Byte Count 1 ( RBCR1 )
0CH	Receive Status Register ( RSR )	Receive Configuration Register ( RCR )

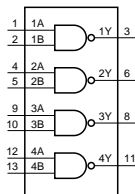
# Philips SAA7114H Video Decoder



# SAA7114H Registers, page 1 of 7 (!)

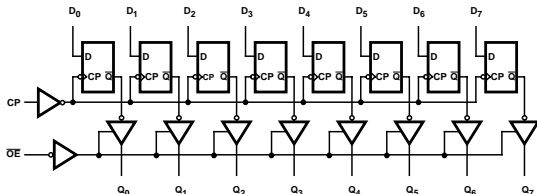
REGISTER FUNCTION	SUB ADDR. (HEX)	D7	D6	D5	D4	D3	D2	D1	D0
<b>Chip version: register 00H</b>									
Chip version (read only)	00	ID07	ID06	ID05	ID04	-	-	-	-
<b>Video decoder: registers 01H to 2FH</b>									
FRONT-END PART: REGISTERS 01H TO 05H									
Horizontal increment delay	01	(1)	(1)	(1)	(1)	IDEL3	IDEL2	IDEL1	IDEL0
Analog input control 1	02	FUSE1	FUSE0	GUDL1	GUDL0	MODE3	MODE2	MODE1	MODE0
Analog input control 2	03	(1)	HLNRS	VBSL	WPOFF	HOLDG	GAFIX	GAI28	GAI18
Analog input control 3	04	GAI17	GAI16	GAI15	GAI14	GAI13	GAI12	GAI11	GAI10
Analog input control 4	05	GAI27	GAI26	GAI25	GAI24	GAI23	GAI22	GAI21	GAI20
DECODER PART: REGISTERS 06H TO 2FH									
Horizontal sync start	06	HSB7	HSB6	HSB5	HSB4	HSB3	HSB2	HSB1	HSB0
Horizontal sync stop	07	HS7	HSS6	HSS5	HSS4	HSS3	HSS2	HSS1	HSS0
Sync control	08	AUFD	FSEL	FOET	HTC1	HTC0	HPLL	VNO11	VNO10
Luminance control	09	BYPS	YCOMB	LDEL	LUBW	LUF13	LUF12	LUF11	LUF10
Luminance brightness control	0A	DBR17	DBR16	DBR15	DBR14	DBR13	DBR12	DBR11	DBR10
Luminance contrast control	0B	DCON7	DCON6	DCON5	DCON4	DCON3	DCON2	DCON1	DCON0
Chrominance saturation control	0C	DSAT7	DSAT6	DSAT5	DSAT4	DSAT3	DSAT2	DSAT1	DSAT0
Chrominance hue control	0D	HUEC7	HUEC6	HUEC5	HUEC4	HUEC3	HUEC2	HUEC1	HUEC0
Chrominance control 1	0E	CDTO	CSTD2	CSTD1	CSTD0	DCVF	FCTC	(1)	CCOMB
Chrominance gain control	0F	ACGC	CGAIN6	CGAIN5	CGAIN4	CGAIN3	CGAIN2	CGAIN1	CGAIN0
Chrominance control 2	10	OFFU1	OFFU0	OFFV1	OFFV0	CHBW	LCBW2	LCBW1	LCBW0
Mode/delay control	11	COLO	RTP1	HDEL1	HDEL0	RTP0	YDEL2	YDEL1	YDEL0
RT signal control	12	RTSE13	RTSE12	RTSE11	RTSE10	RTSE03	RTSE02	RTSE01	RTSE00
RT/X-port output control	13	RTCE	XRHS	XRV51	XRV50	HLSEL	OFTS2	OFTS1	OFTS0
Analog/ADC/compatibility control	14	CM99	UPTCV	AOSL1	AOSL0	XTOUTE	OLDSB	APCK1	APCK0
VGATE start, FID change	15	VSTA7	VSTA6	VSTA5	VSTA4	VSTA3	VSTA2	VSTA1	VSTA0
VGATE stop	16	VSTO7	VSTO6	VSTO5	VSTO4	VSTO3	VSTO2	VSTO1	VSTO0
Miscellaneous/VGATE MSBs	17	LLCE	LLC2E	(1)	(1)	(1)	VGPS	VSTO8	VSTA8

# Fixed-function: The 7400 series



7400

Quad NAND Gate



74374

Octal D Flip-Flop