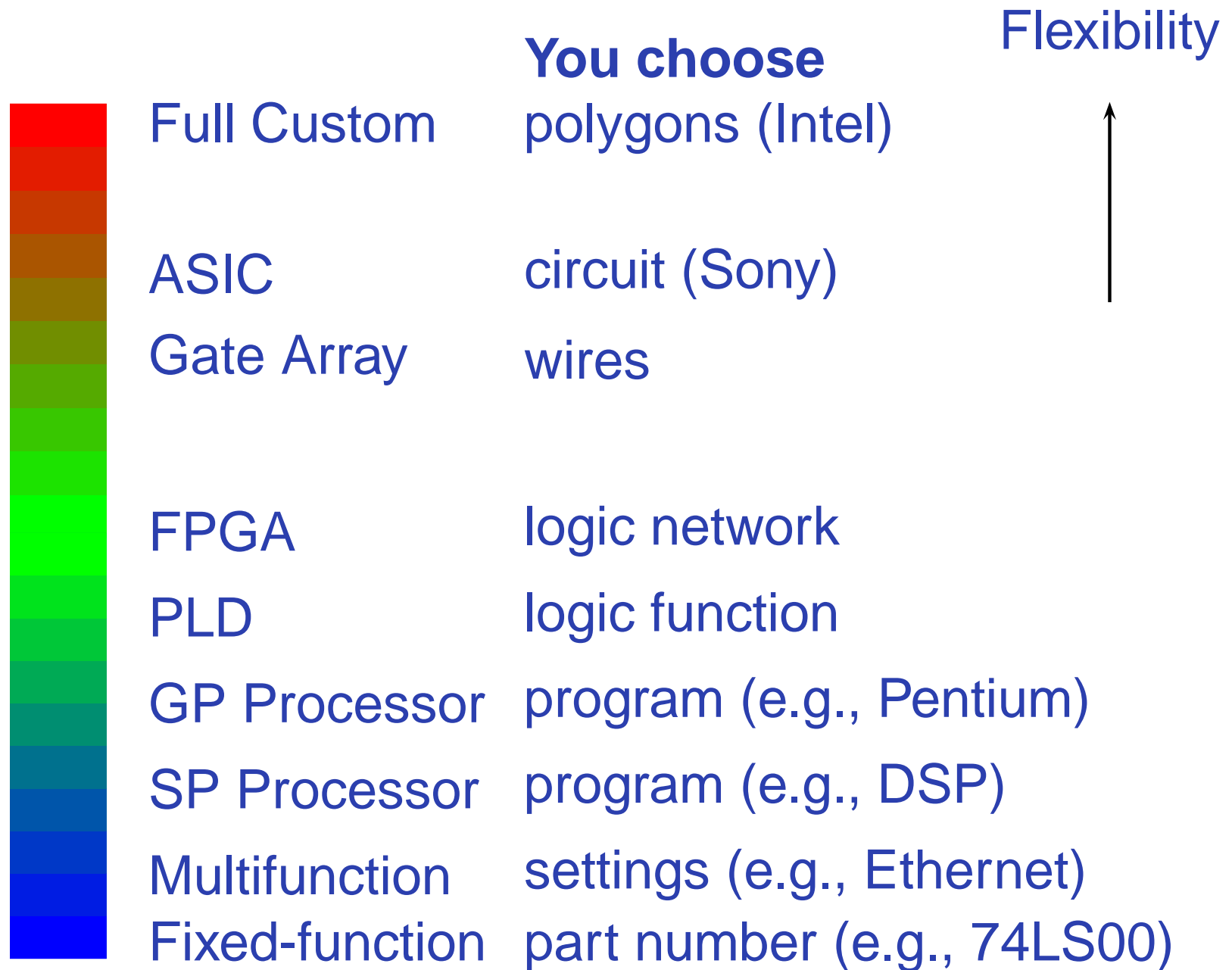# Processors, FPGAs, and ASICs

## *CSEE W4840*
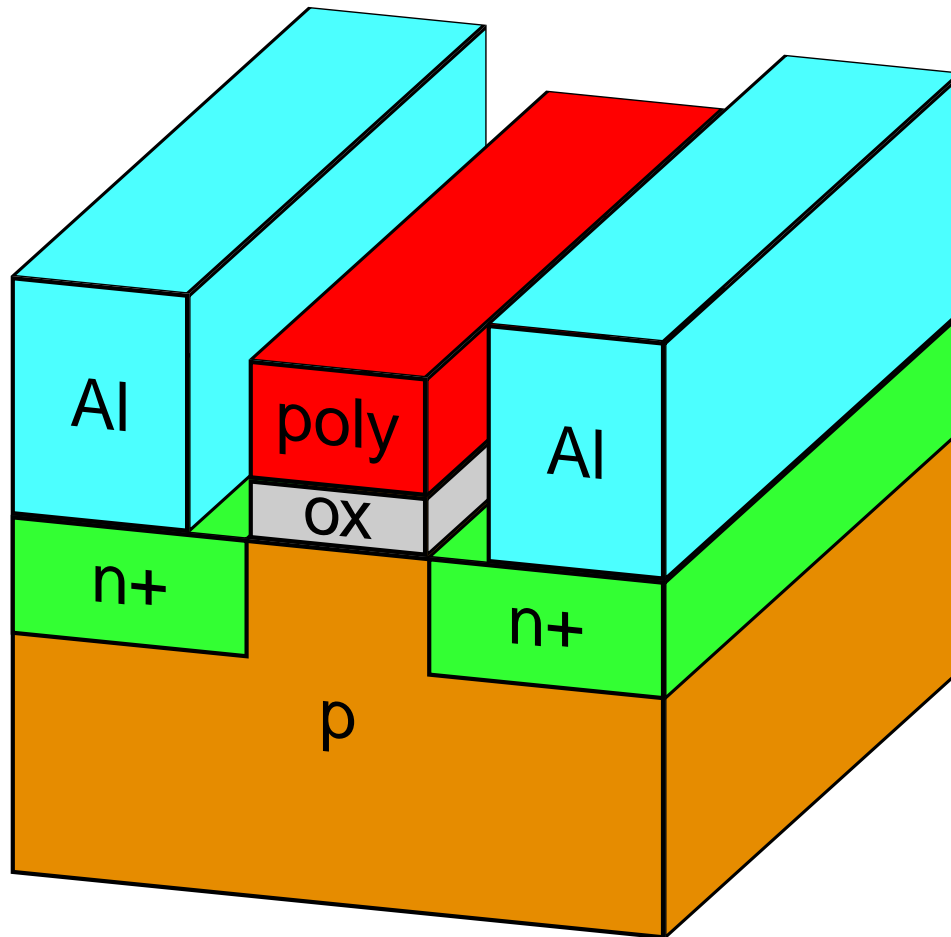
Prof. Stephen A. Edwards

Columbia University
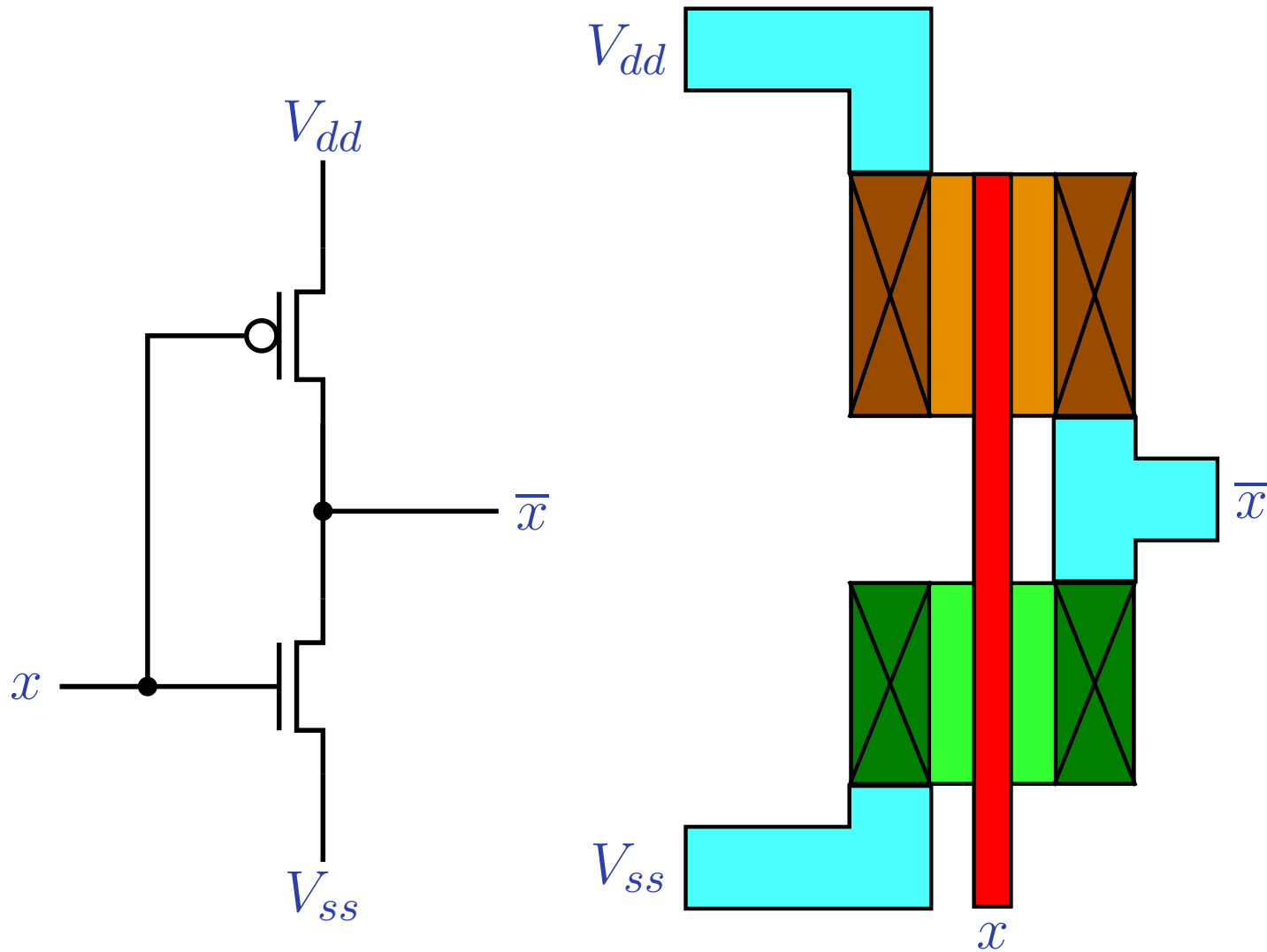
# Spectrum of IC choices

Flexibility

|  | | **You choose** |
|---|---|---|
| Full Custom | | polygons (Intel) |
| ASIC | | circuit (Sony) |
| Gate Array | | wires |
| FPGA | | logic network |
| PLD | | logic function |
| GP Processor | | program (e.g., Pentium) |
| SP Processor | | program (e.g., DSP) |
| Multifunction | | settings (e.g., Ethernet) |
| Fixed-function | | part number (e.g., 74LS00) |

# Inverter Transistors and Layout
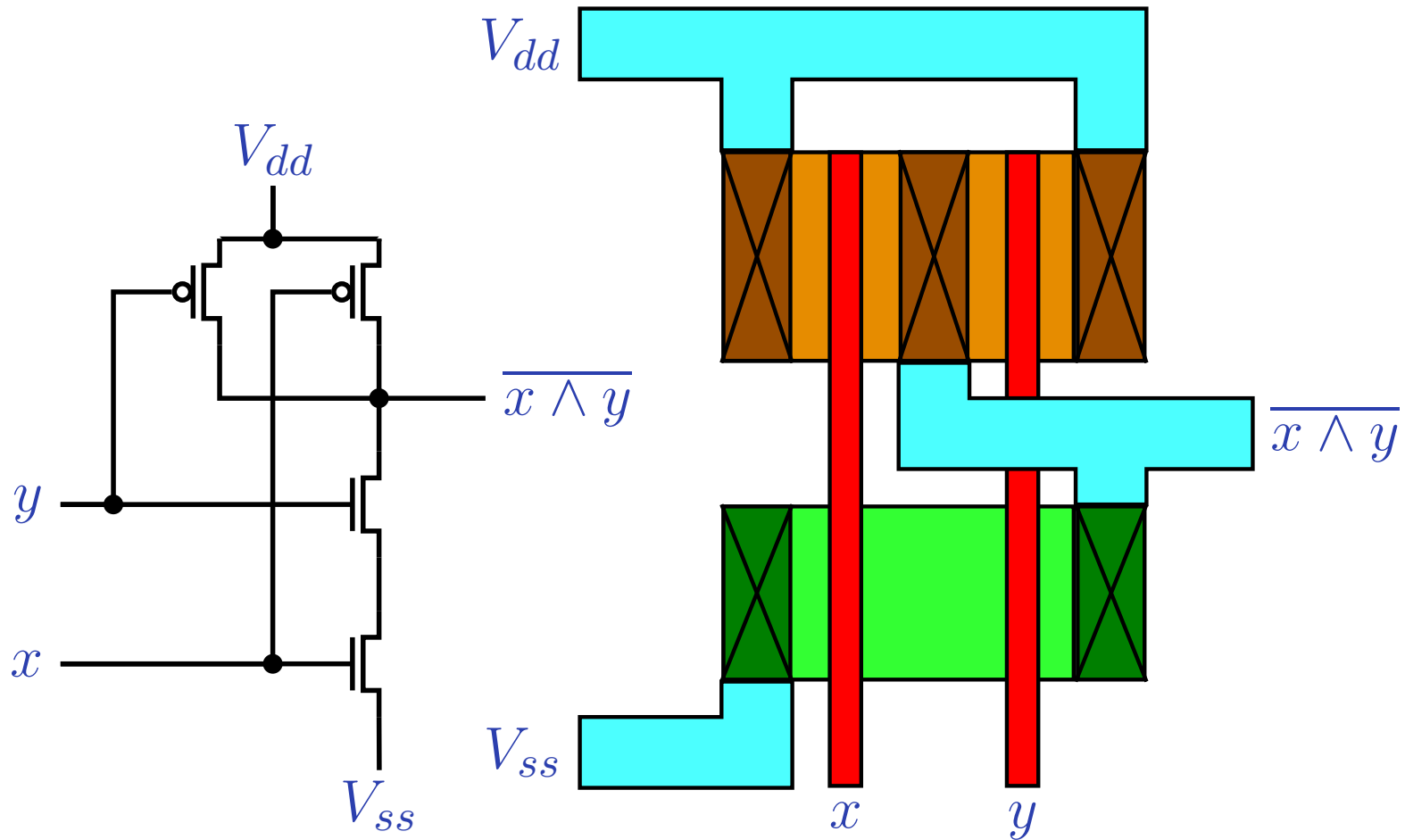
# Full-custom ICs
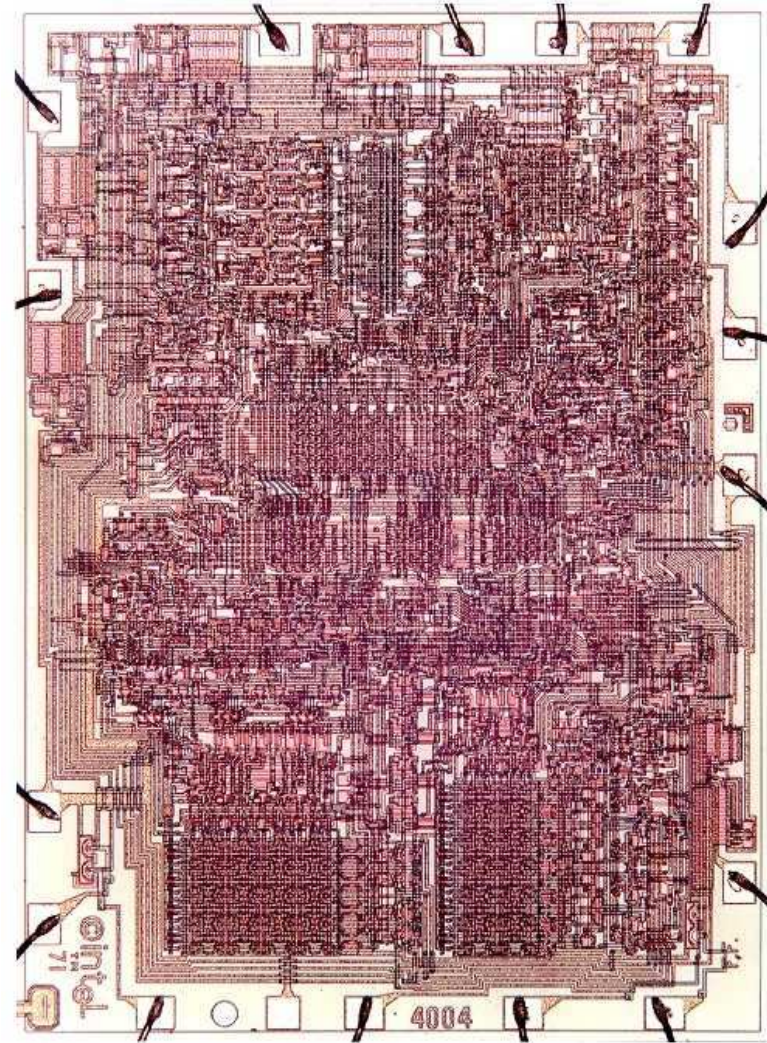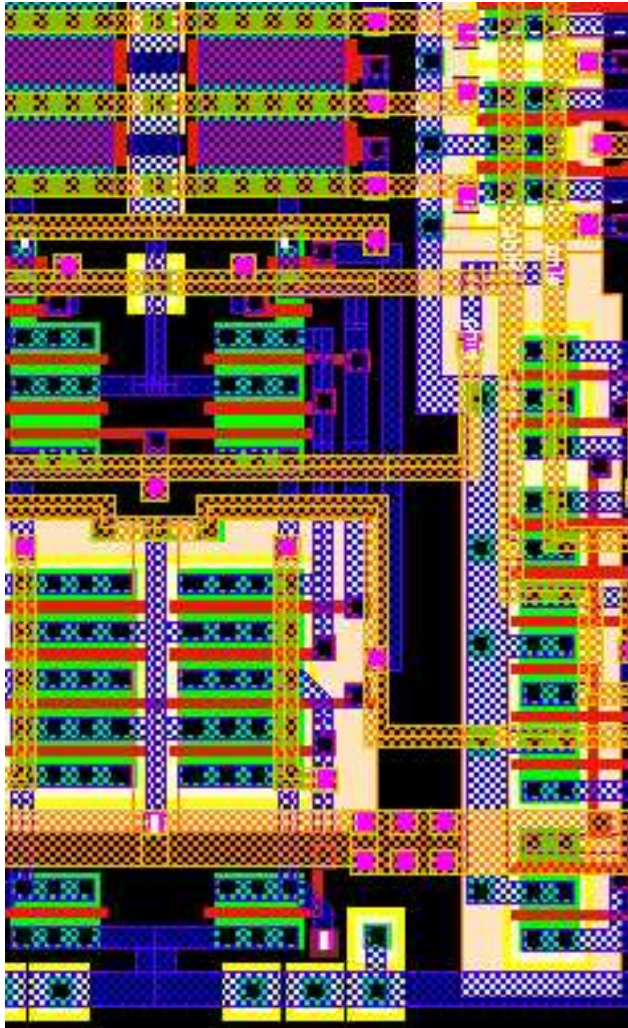
# Standard Cell ASICs
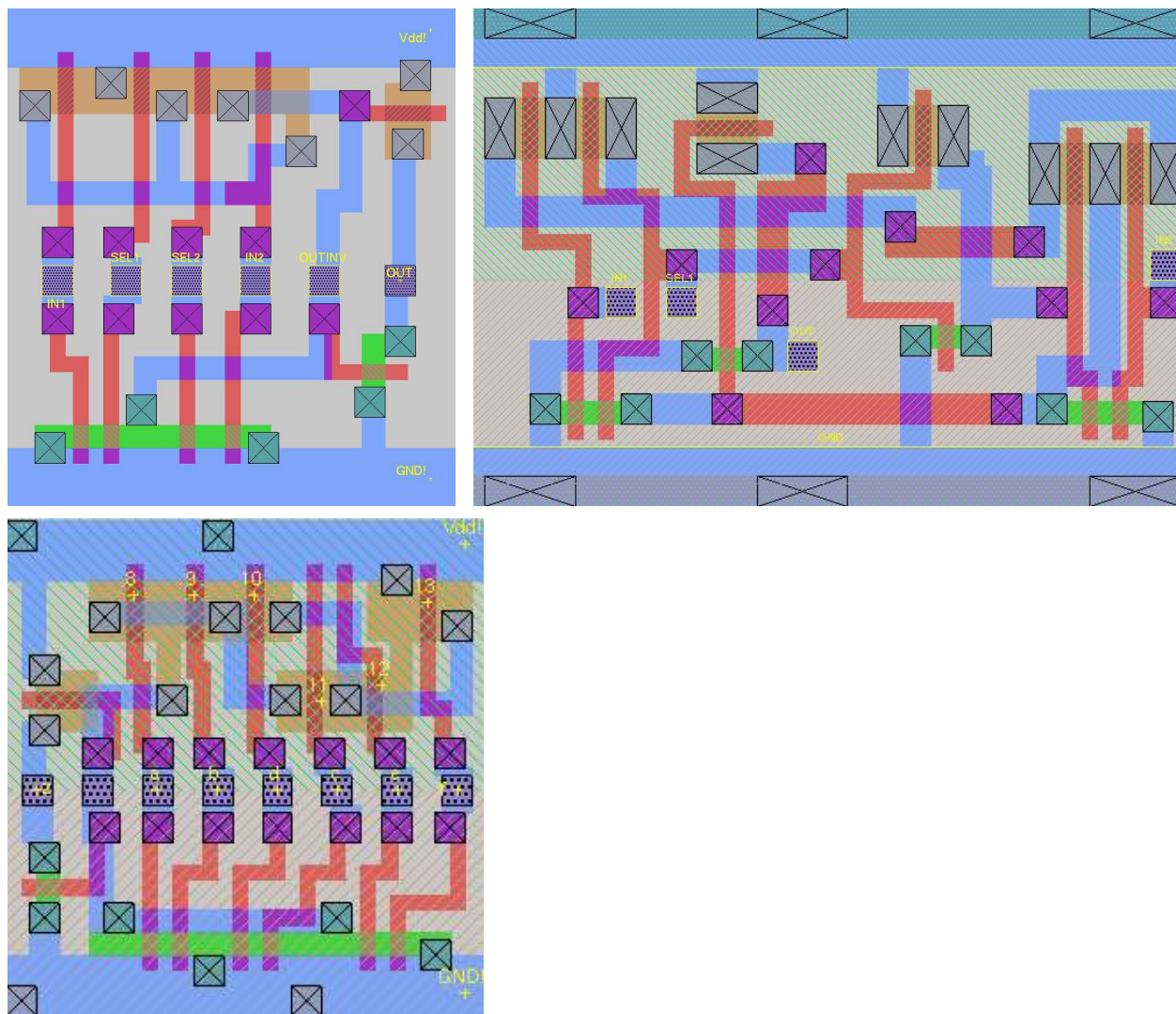
# Standard Cell ASICs
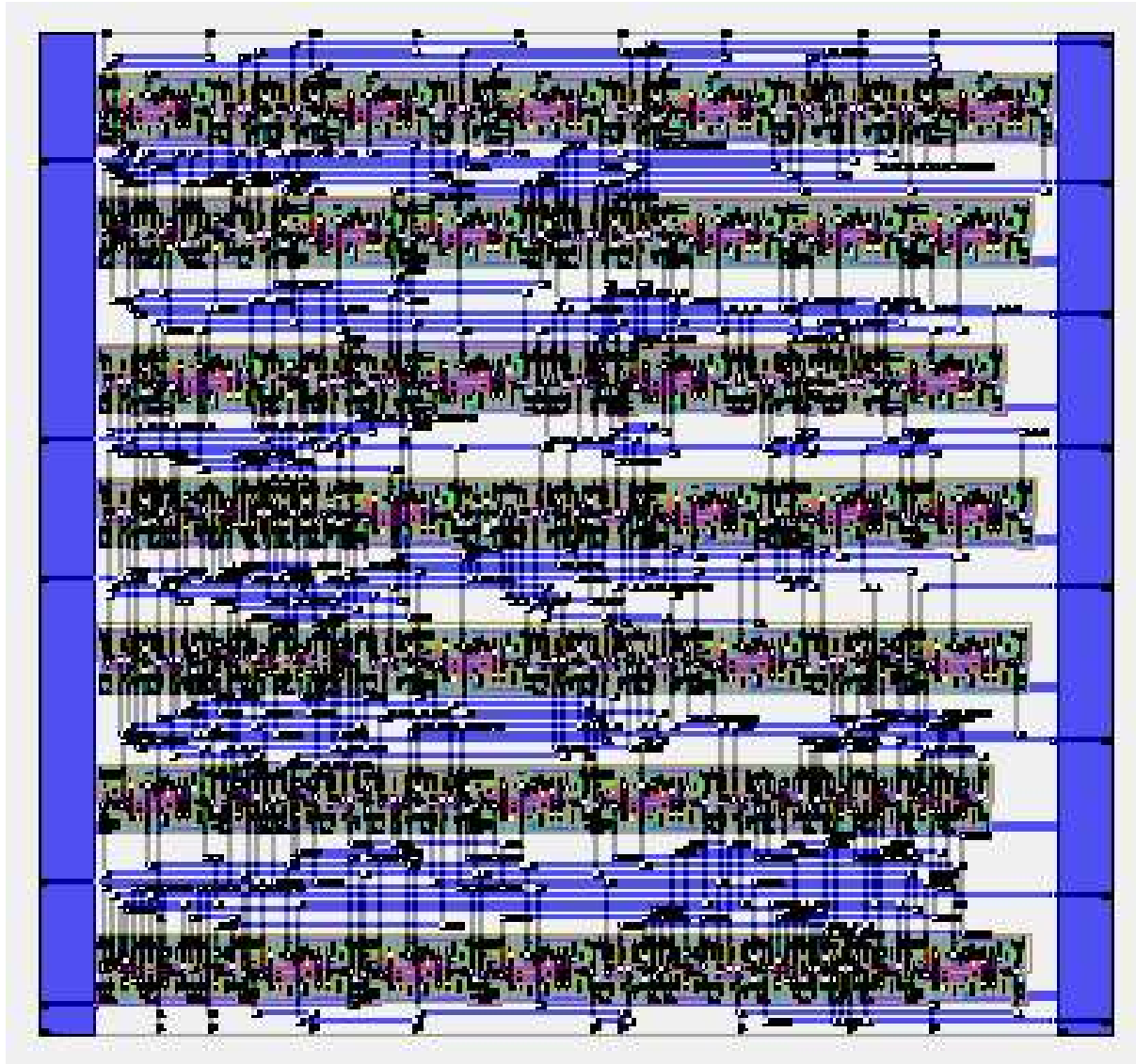
# Channeled Gate Arrays
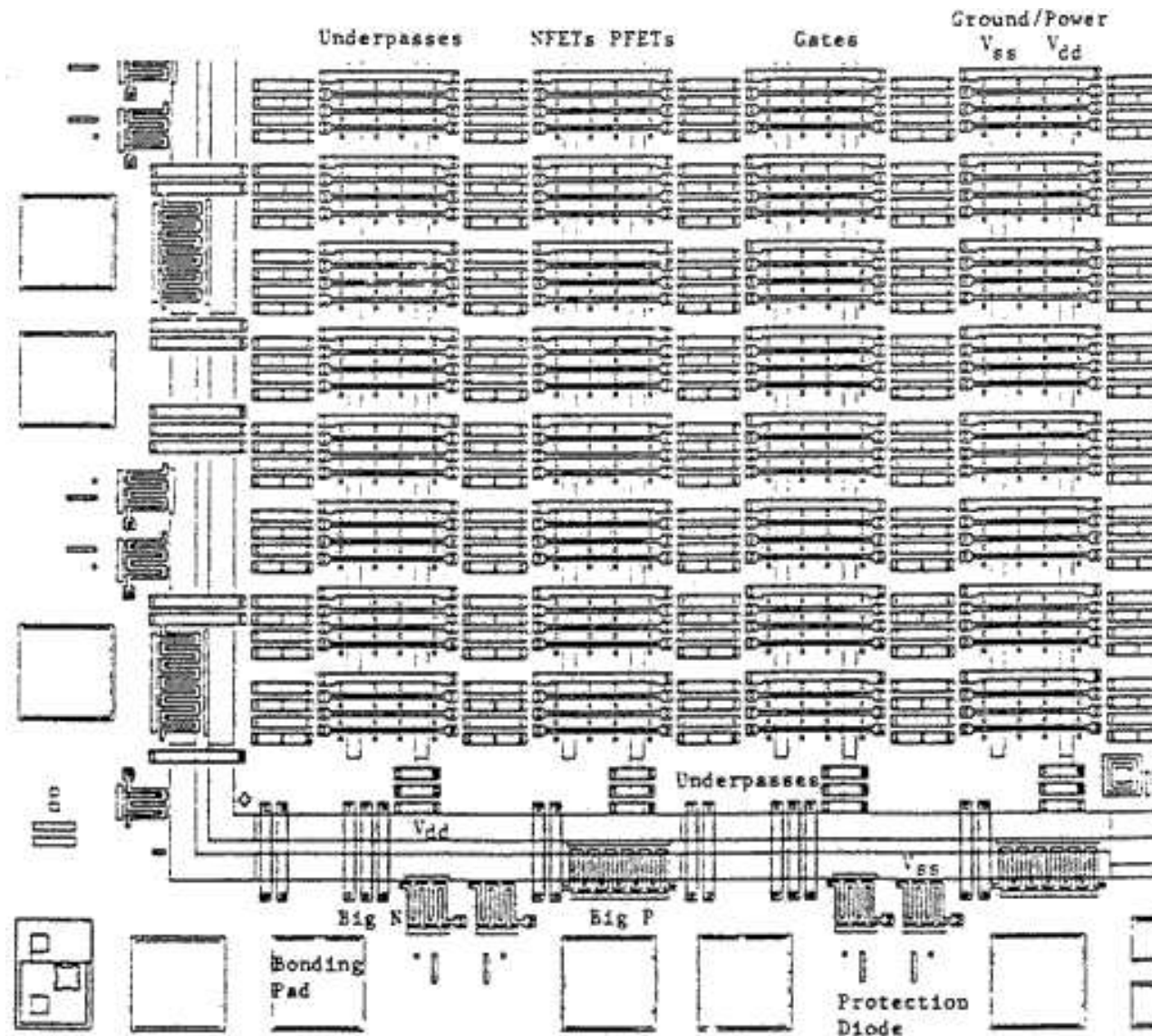
# Channeled Gate Arrays
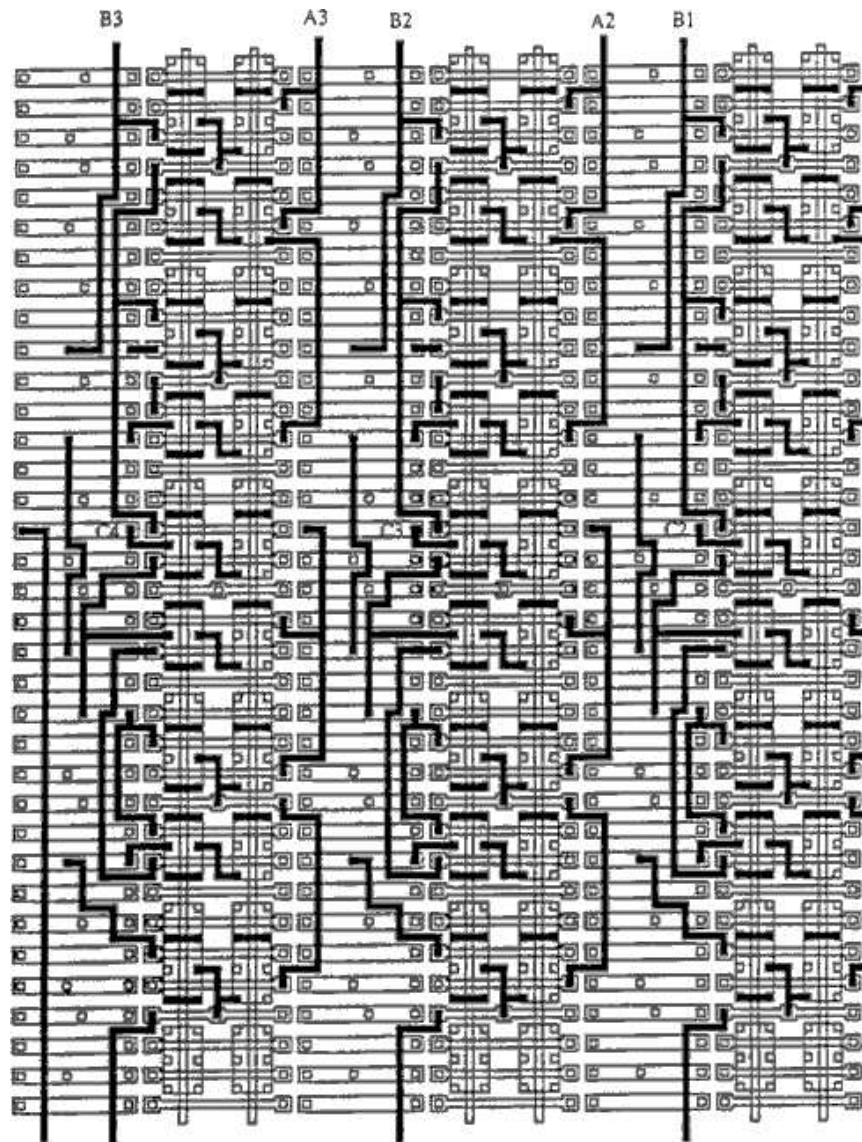
# Sea-of-Gates Gate Arrays

# FPGAs: Floorplan



DLL

BLOCK RAM

CLBs

CLBs

BLOCK RAM

DLL

BLOCK RAM

CLBs

CLBs

BLOCK RAM

DLL

I/O LOGIC

# FPGAs: Routing
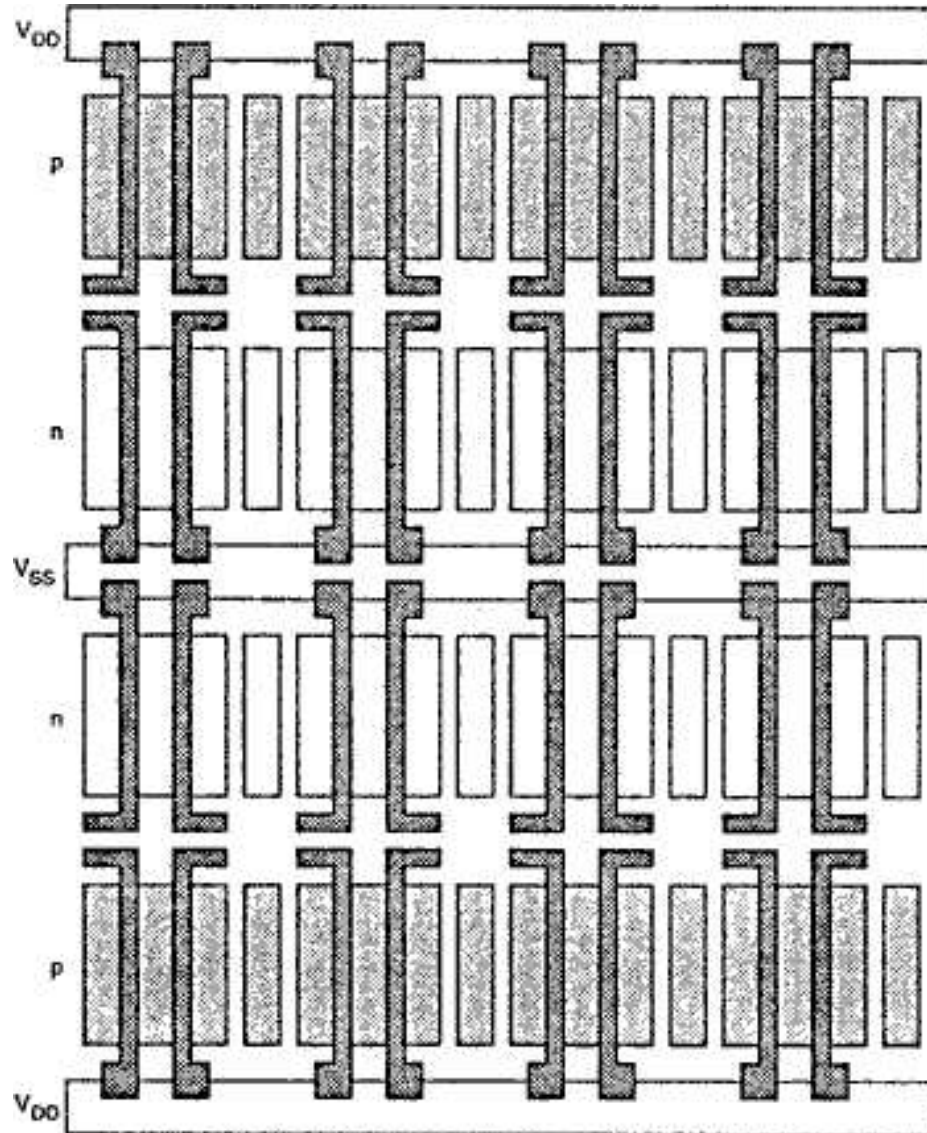


Single-length line Switch Matrix connections
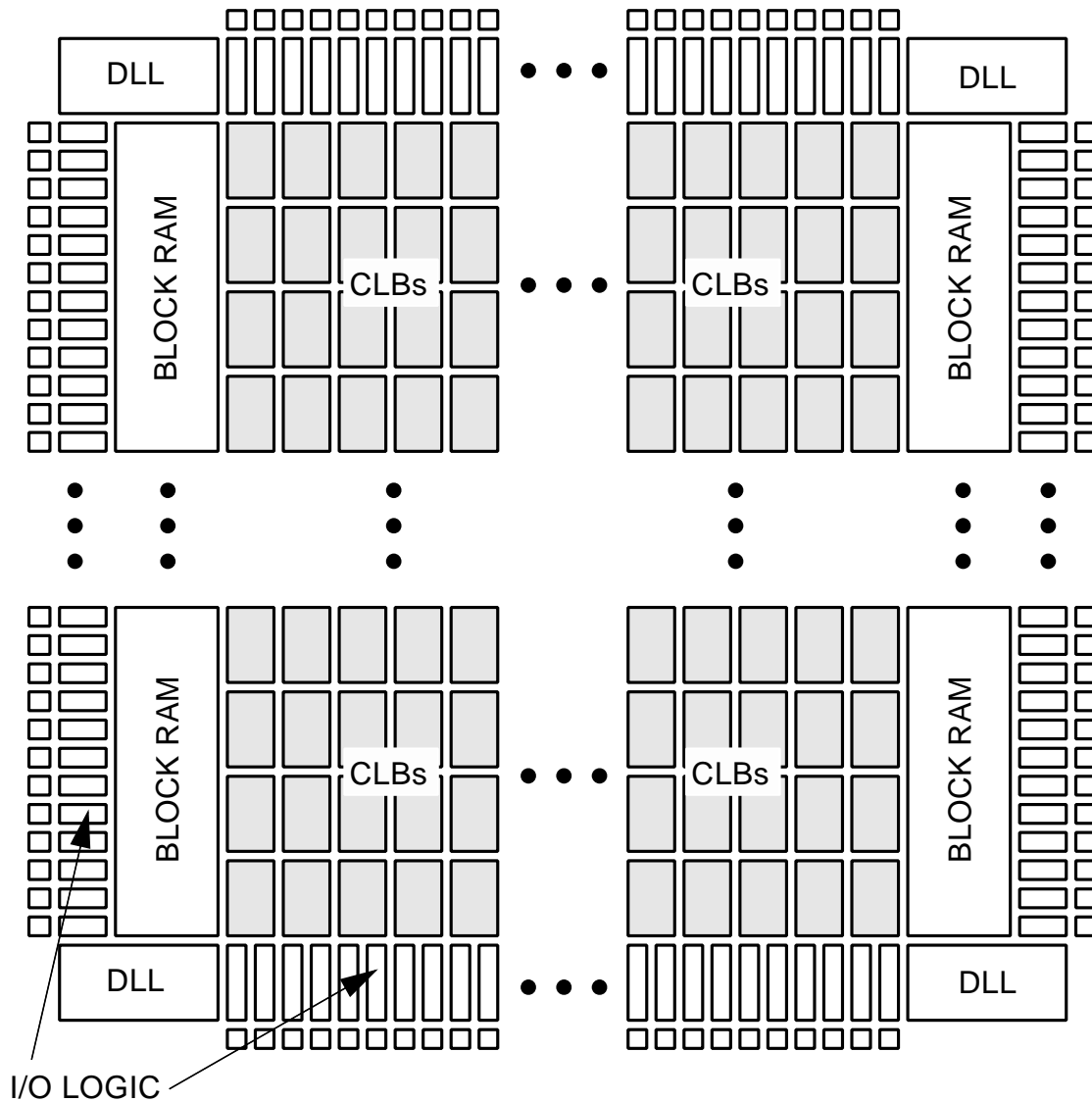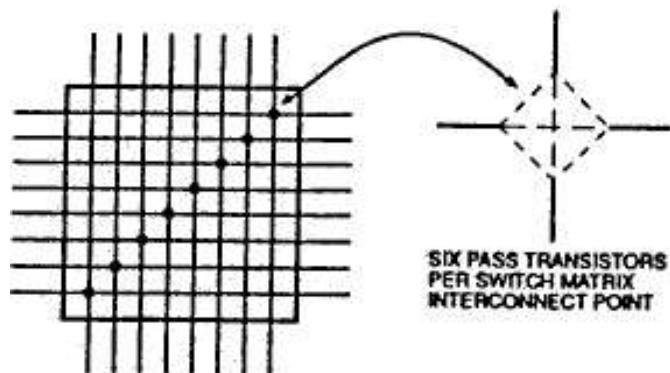
SIX PASS TRANSISTORS
PER SWITCH MATRIX
INTERCONNECT POINT



SWITCH MATRICES
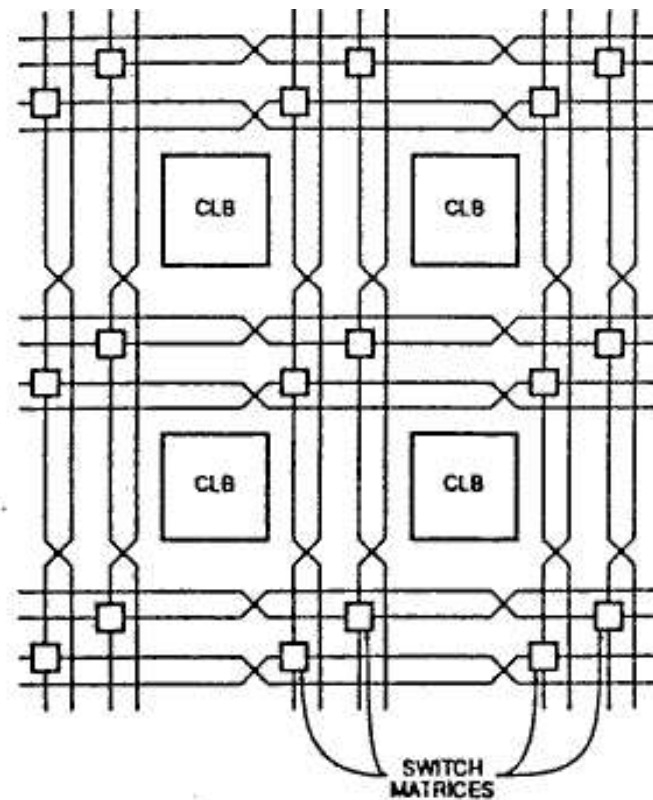
Double-length lines in CLB array

# FPGAs: CLB
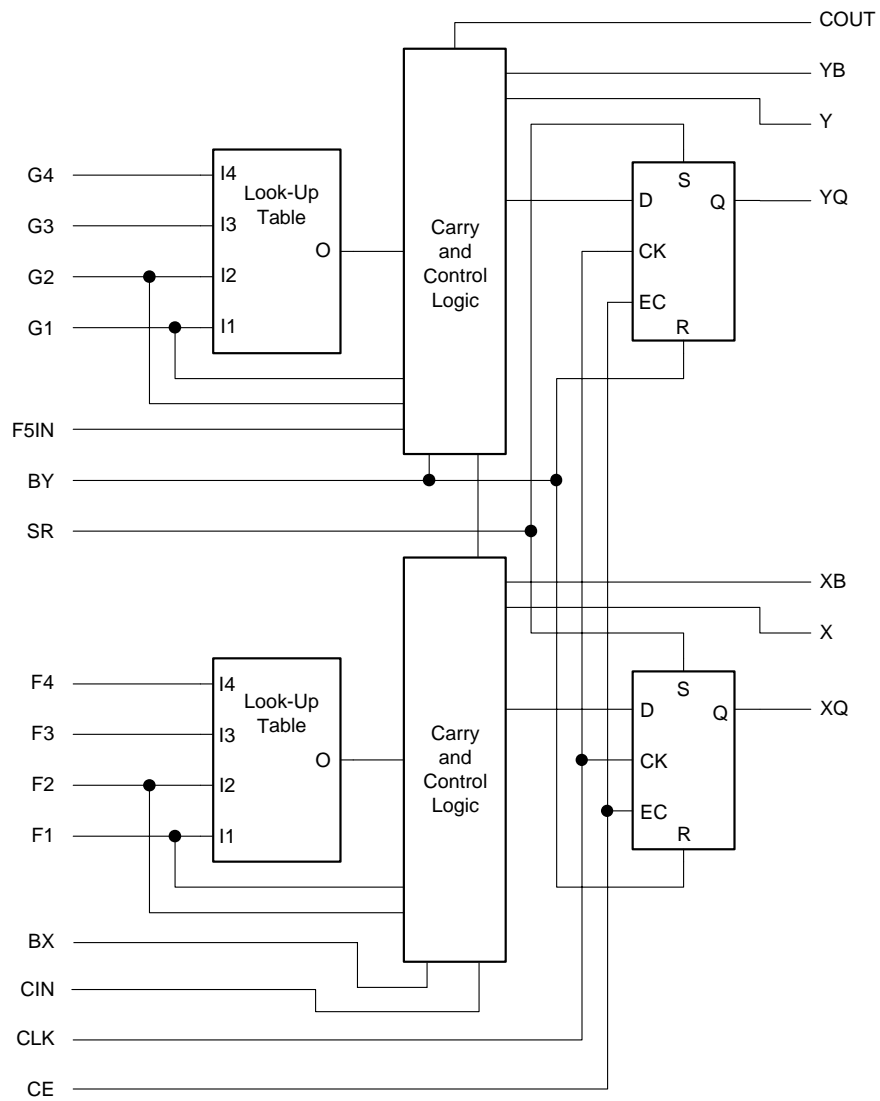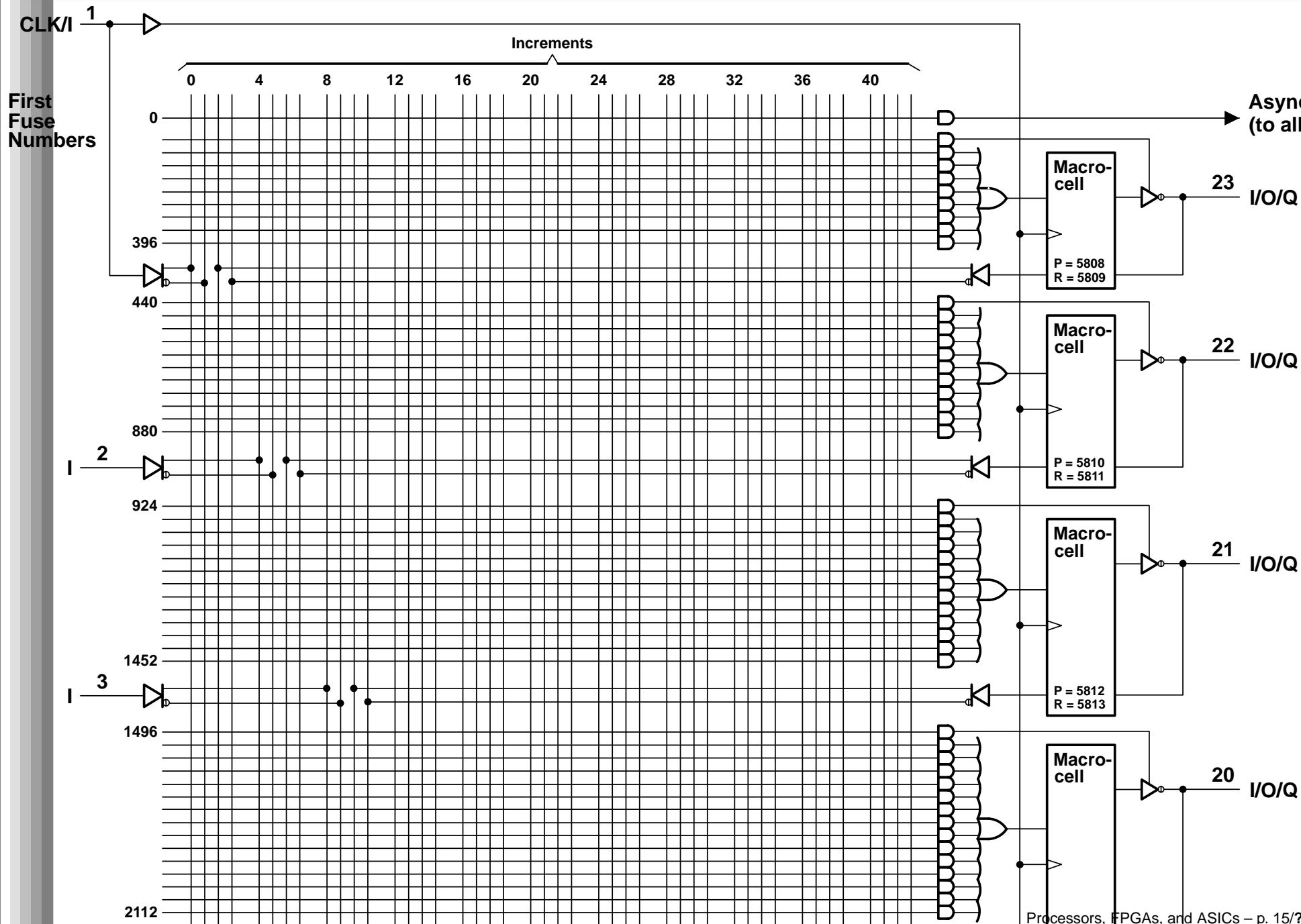
# Example: Euclid's Algorithm

```c
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

| r0 | Always 0 |
| r1 | Global Registers |
| ⋮ | |
| r7 | |
| r8/o0 | Output Registers |
| ⋮ | |
| r14/o6 | Stack Pointer |
| r15/o7 | |
| r16/l0 | Local Registers |
| ⋮ | |
| r23/l7 | |

31    0

| r24/i0 | Input Registers |
| ⋮ | |
| r30/i6 | Frame Pointer |
| r31/i7 | Return Address |

| PSW | Status Word |
| PC | Program Counter |
| nPC | Next PC |
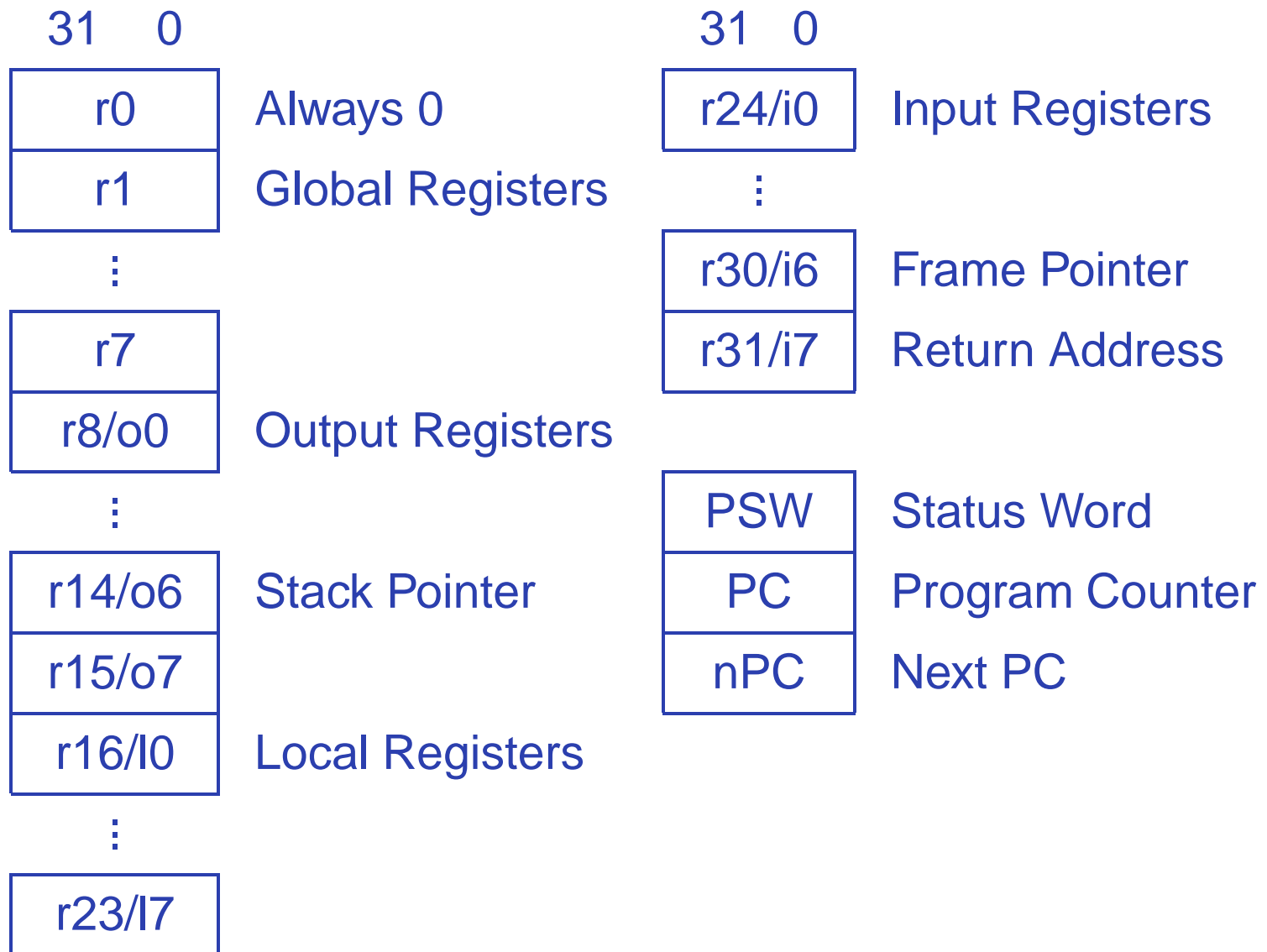
# SPARC Register Windows

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

| | | r8/o0 |
|---|---|---|
| | | ⋮ |
| | | r15/o7 |
| | | r16/l0 |
| | | ⋮ |
| | | r23/l7 |
| | r8/o0 | r24/i0 |
| | ⋮ | ⋮ |
| | r15/o7 | r31/i7 |
| | r16/l0 | |
| | ⋮ | |
| | r23/l7 | |
| r8/o0 | r24/i0 | |
| ⋮ | ⋮ | |
| r15/o7 | r31/i7 | |
| r16/l0 | | |
| ⋮ | | |
| r23/l7 | | |
| r24/i0 | | |
| ⋮ | | |
| r31/i7 | | |
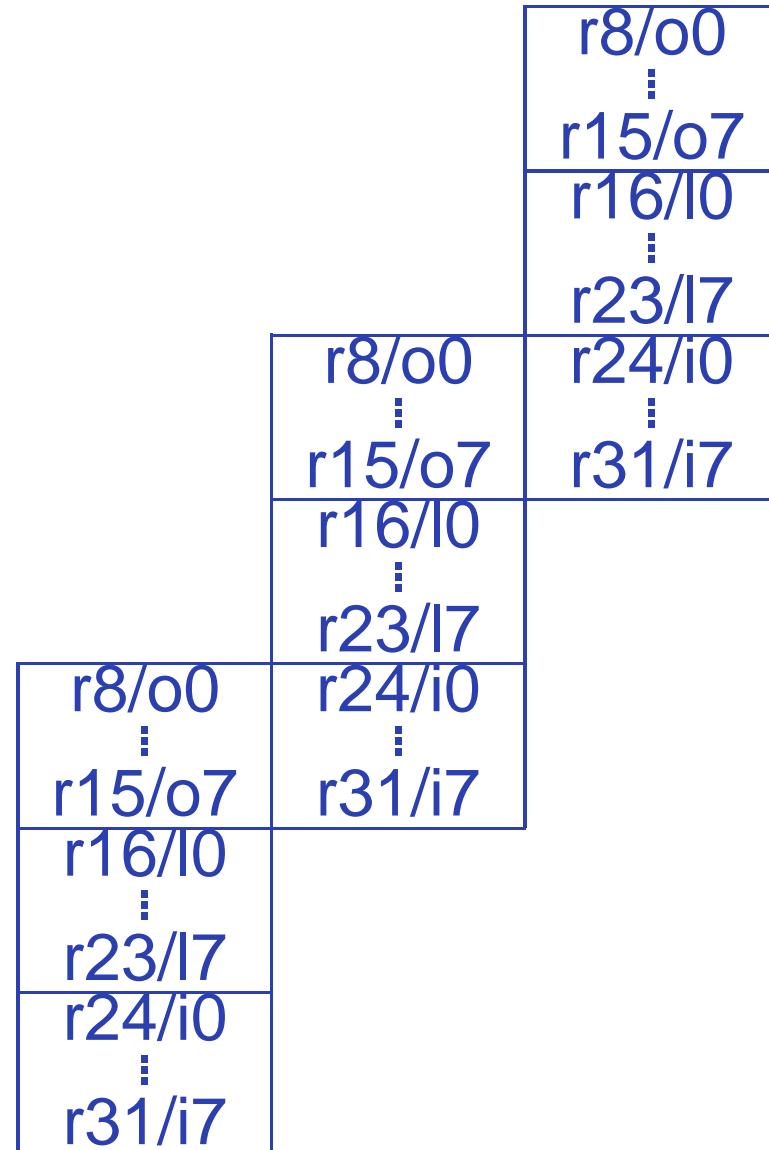
# Euclid on the SPARC

```
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
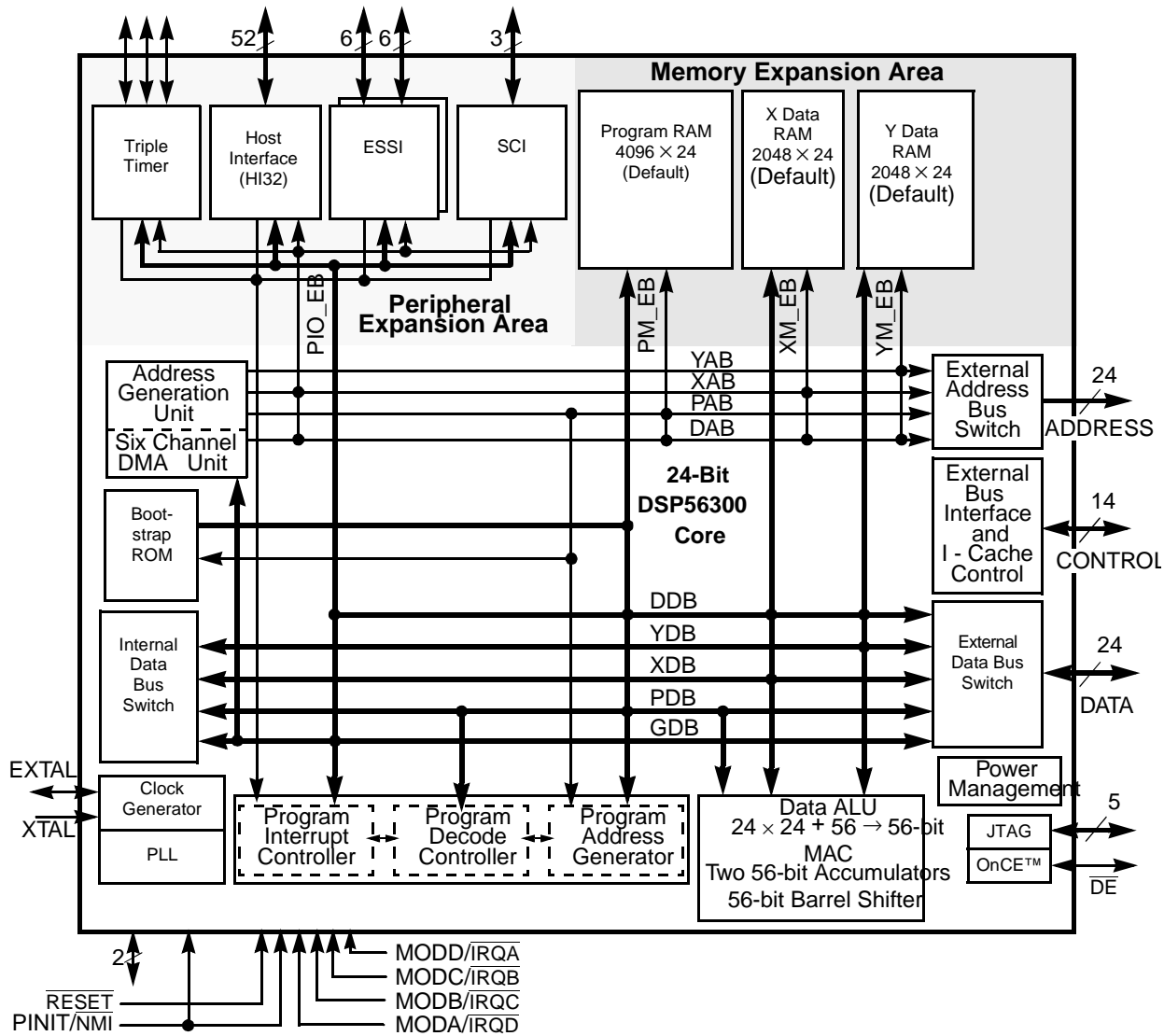
# DSP 56000 Programmer's Model

```
  55 4847    2423     0              15     0
        +--------+--------+                +--------+
        |   x1   |   x0   | Source        |        | Program Counter
        +--------+--------+ Registers     +--------+
        |   y1   |   y0   |               |        | Status Register
   +----+--------+--------+               +--------+
   | a2 |   a1   |   a0   | Accumulator   |        | Loop Address
   +----+--------+--------+               +--------+
   | b2 |   b1   |   b0   | Accumulator   |        | Loop Count
   +----+--------+--------+               +--------+

  15    0 15    0 15      0              +--------+
   +------+ +------+ +------+        15   |        | PC Stack
   |  r7  | |  n7  | |  m7  |             +--------+
   +------+ +------+ +------+              :
      :        :        :             0   +--------+
   +------+ +------+ +------+              |        |
   |  r4  | |  n4  | |  m4  | Address      +--------+
   +------+ +------+ +------+
   |  r3  | |  n3  | |  m3  | Registers   15  +--------+ SR Stack
   +------+ +------+ +------+                 |        |
      :        :        :                     +--------+
   +------+ +------+ +------+                  :
   |  r0  | |  n0  | |  m0  |            0    +--------+
   +------+ +------+ +------+                 |        |
                                             +--------+

                                       +--------+
                                       |        | Stack pointer
                                       +--------+
```
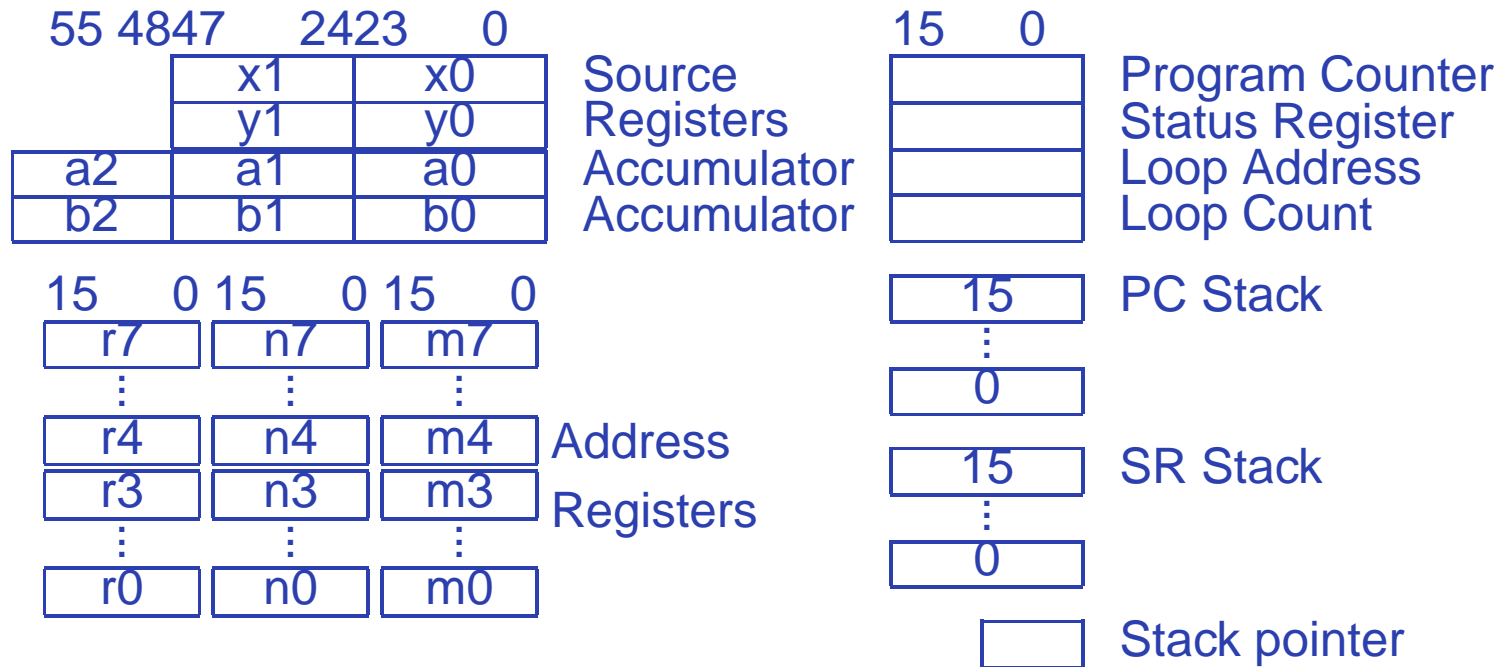
# Motorola DSP56301 ALU

# Motorola DSP56301 AGU

Low Address ALU ← → High Address ALU

XAB   YAB   PAB

Triple Multiplexer

| N0 | M0 |
|----|----|
| N1 | M1 |
| N2 | M2 |
| N3 | M3 |

Address ALU

| EP | |
|----|----|
| R0 | R4 |
| R1 | R5 |
| R2 | R6 |
| R3 | R7 |

Address ALU

| M4 | N4 |
|----|----|
| M5 | N5 |
| M6 | N6 |
| M7 | N7 |

Global Data Bus

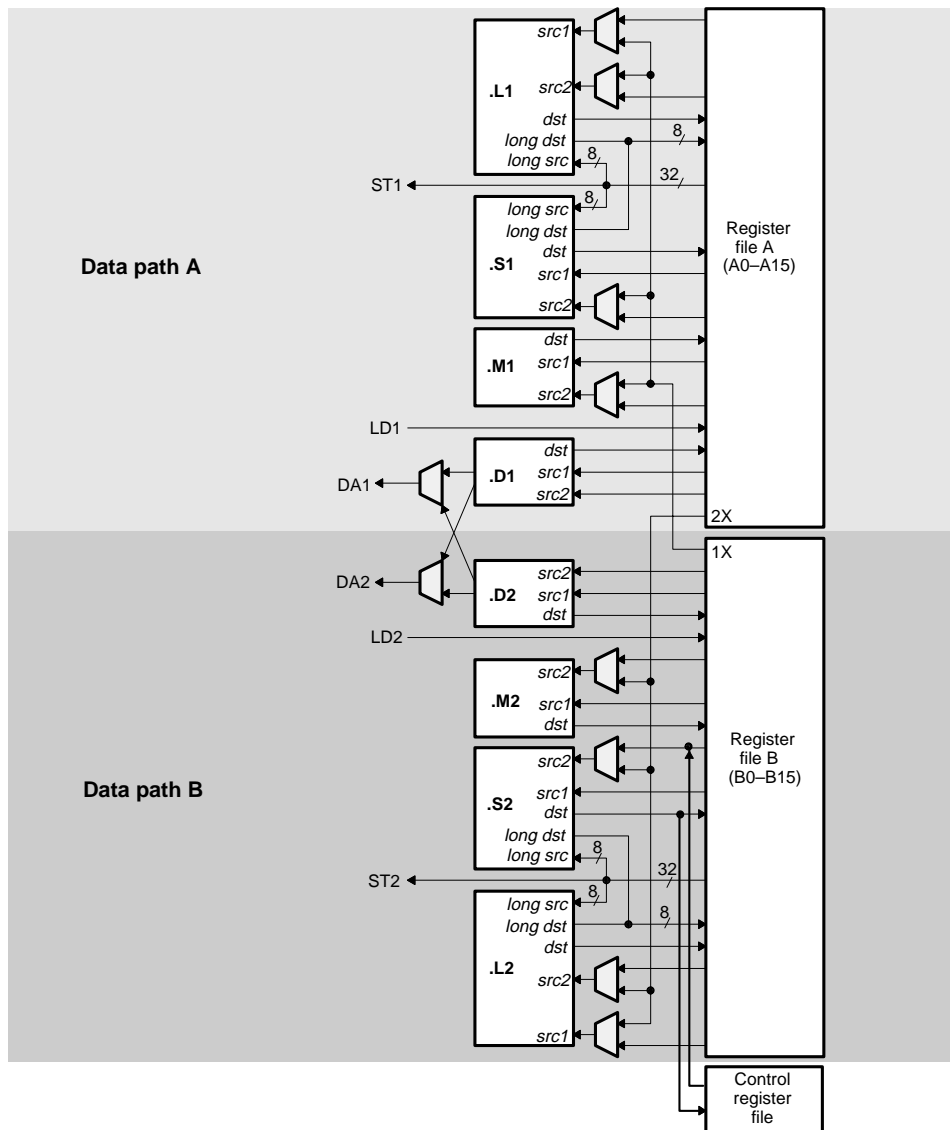Program Address Bus
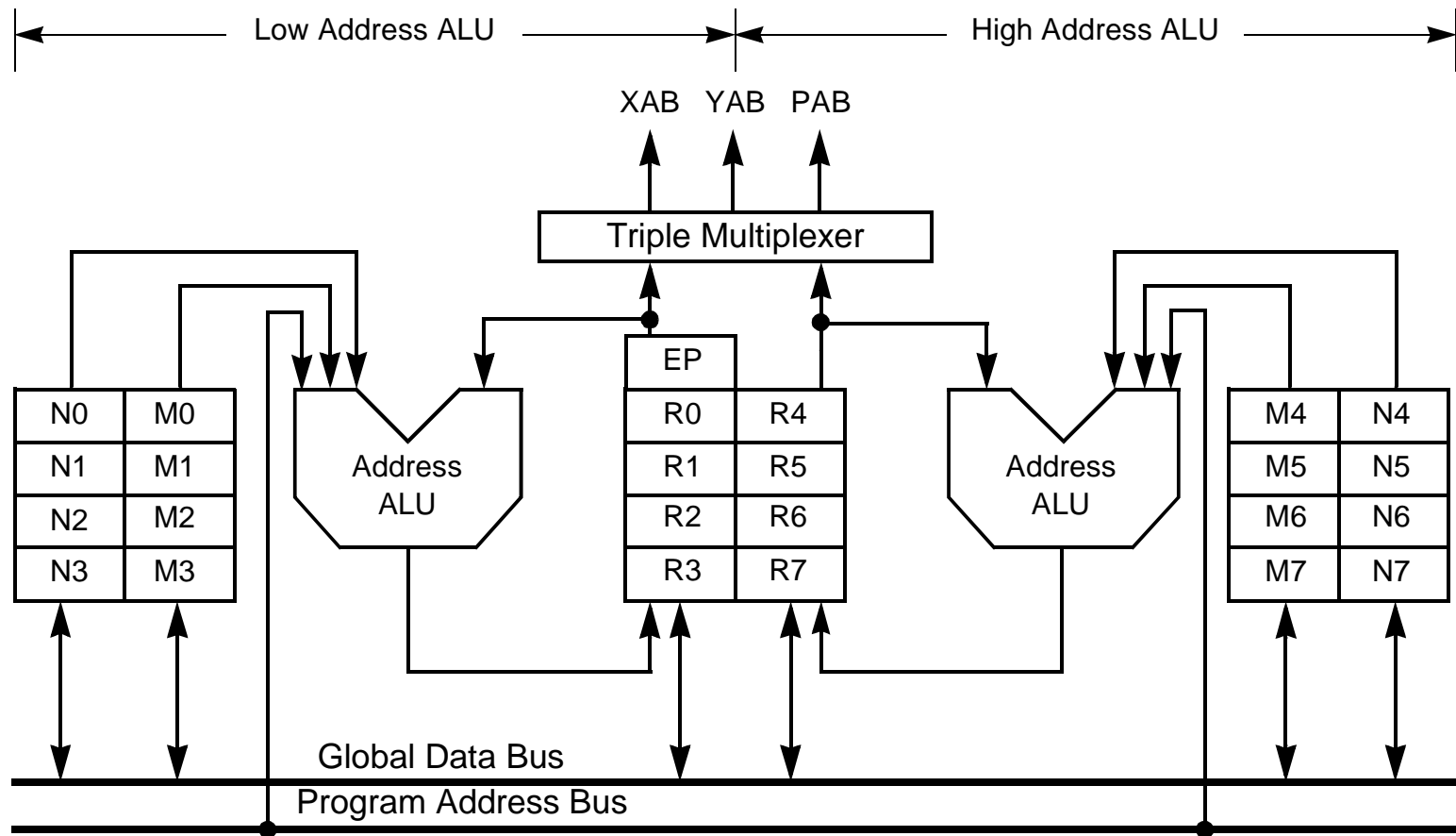
# FIR Filter in 56000

```
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
```

Load a halfword (16 bits)

Do this on unit D1

```
FIRLOOP:
        LDH  .D1  *A1++, A2    ; Fetch next sample
||      LDH  .D2  *B1++, B2    ; Fetch next coeff.
|| [B0] SUB  .L2  B0, 1, B0    ; Decrement count
|| [B0] B    .S2  FIRLOOP      ; Branch if non-zero
||      MPY  .M1X A2, B2, A3   ; Sample × Coeff.
||      ADD  .L1  A4, A3, A4   ; Accumulate result
```

Use the cross path

Predicated instruction (only if B0 non-zero)

Run these instruction in parallel

# AX88796 Ethernet Controller
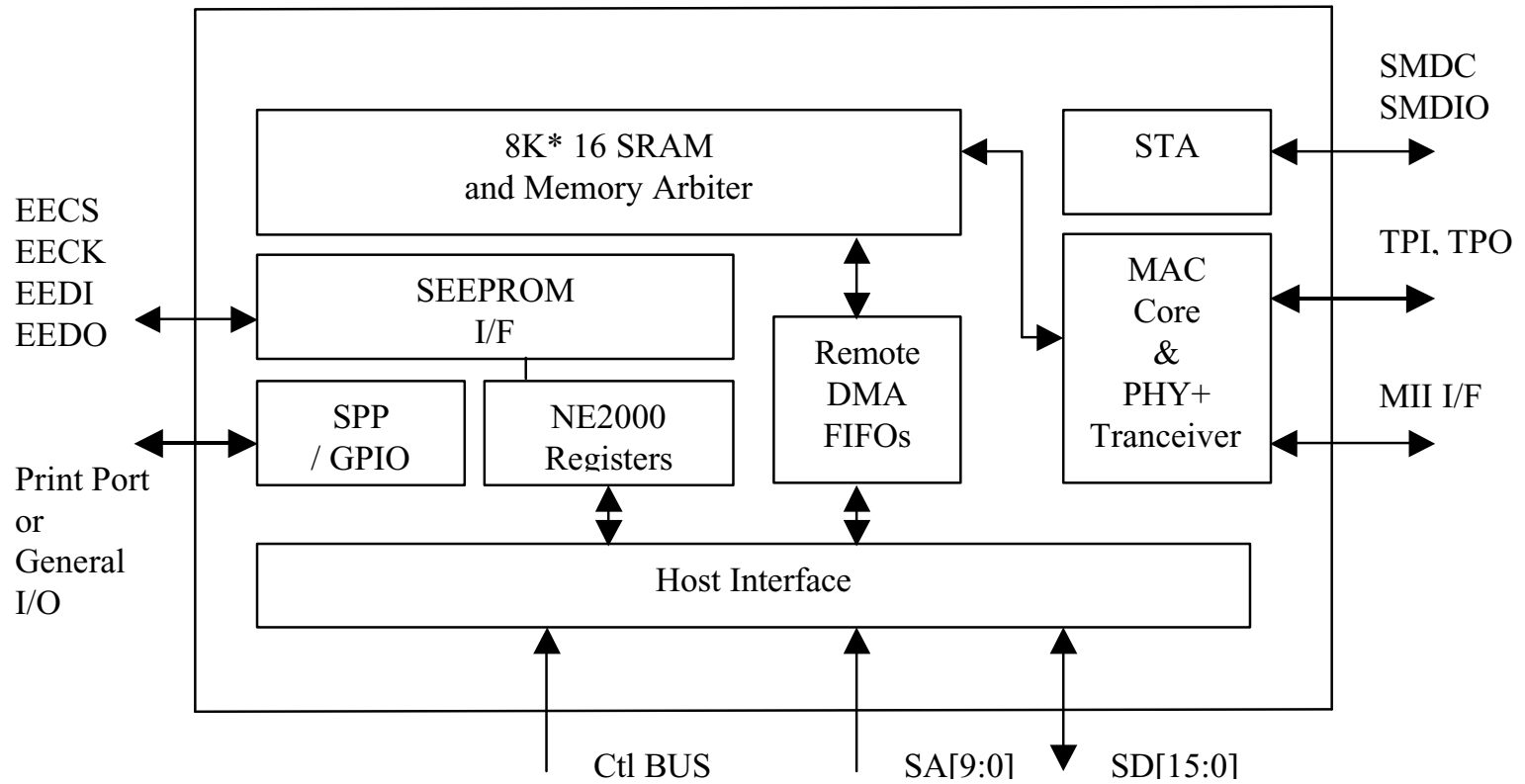
# Ethernet Controller Registers

PAGE 0 (PS1=0,PS0=0)

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

# Philips SAA7114H Video Decoder
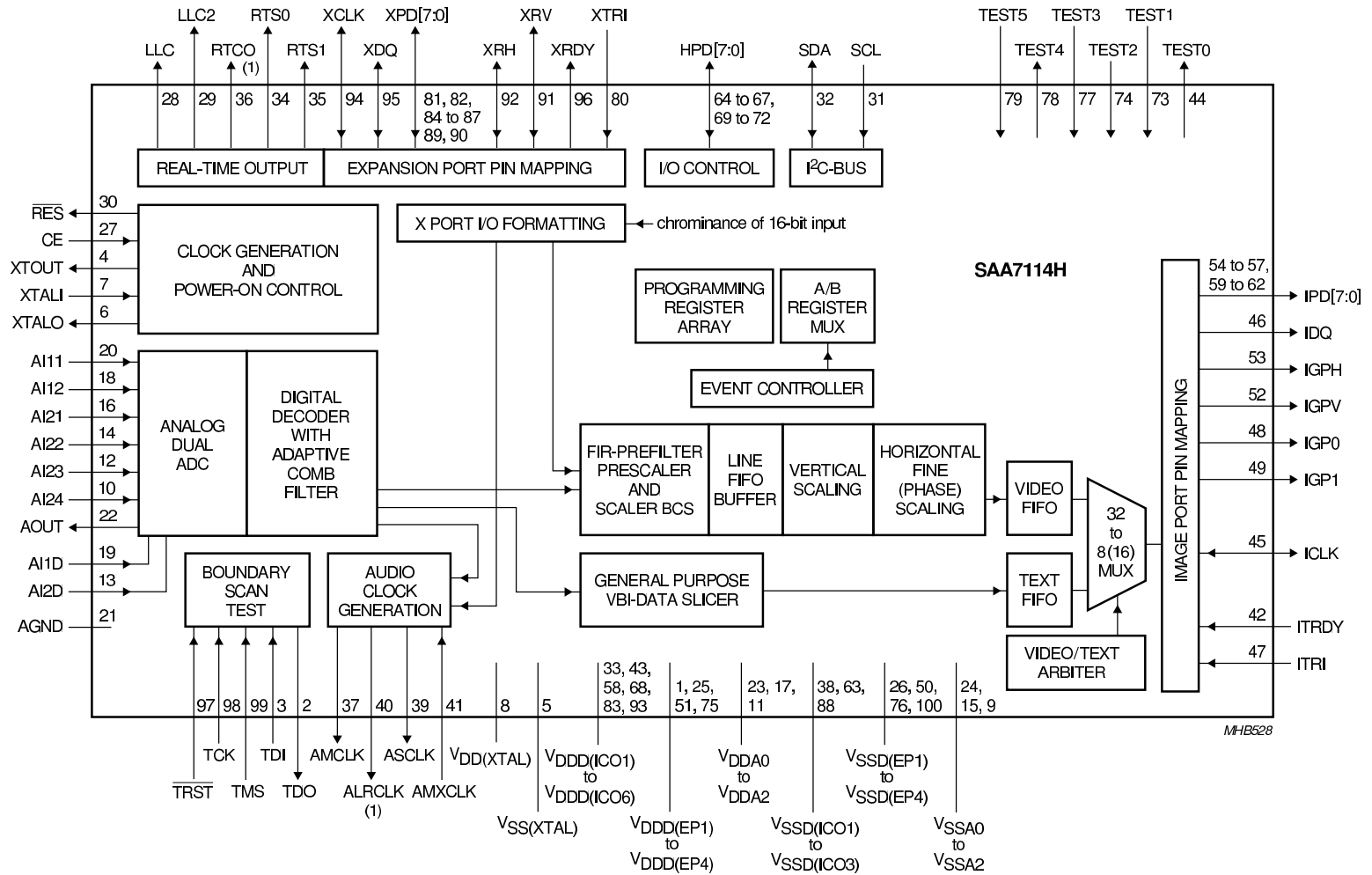
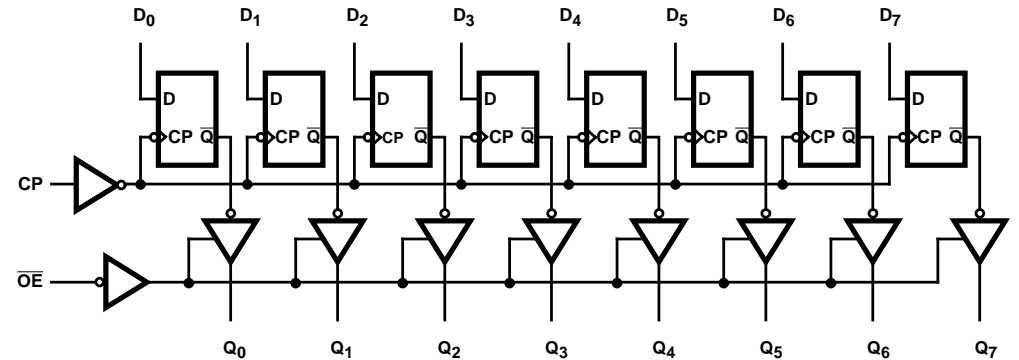| REGISTER FUNCTION | SUB ADDR. (HEX) | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|---|---|
| **Chip version: register 00H** | | | | | | | | | |
| Chip version (read only) | 00 | ID07 | ID06 | ID05 | ID04 | – | – | – | – |
| **Video decoder: registers 01H to 2FH** | | | | | | | | | |
| 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 | HSS7 | HSS6 | HSS5 | HSS4 | HSS3 | HSS2 | HSS1 | HSS0 |
| Sync control | 08 | AUFD | FSEL | FOET | HTC1 | HTC0 | HPLL | VNOI1 | VNOI0 |
| Luminance control | 09 | BYPS | YCOMB | LDEL | LUBW | LUFI3 | LUFI2 | LUFI1 | LUFI0 |
| Luminance brightness control | 0A | DBRI7 | DBRI6 | DBRI5 | DBRI4 | DBRI3 | DBRI2 | DBRI1 | DBRI0 |
| 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 | XRVS1 | XRVS0 | 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
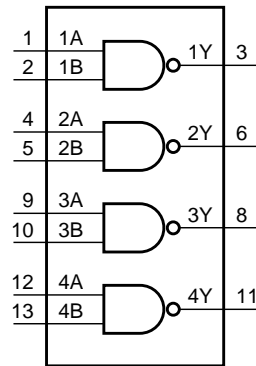
7400

Quad NAND Gate

74374

Octal D Flip-Flop