Processors, FPGAs, and ASICs

Stephen A. Edwards

Columbia University

Spring 2019
Spectrum of IC choices

Flexible, efficient
- Full Custom
- ASIC
- Gate Array
- FPGA
- PLD
- GP Processor
- SP Processor
- Multifunction

You choose
- Polygons (Intel)
- Circuit (Sony)
- Wires
- Logic network
- Logic function
- Program (e.g., ARM)
- Program (e.g., DSP)
- Settings (e.g., Ethernet Ctrl.)
- Part number (e.g., 74HCT00)

Cheap, quick to design
An N-Channel MOS Transistor

Gate at 0V: Off

Gate

SiO₂

Drain

Source

n

p (holes)

n

Ammeter

3 V

0 V

0 V
An N-Channel MOS Transistor

Gate positive: On

SiO₂

Drain

Source

p (holes)

Ammeter

3 V

3 V
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

Diagram showing the layout of FPGAs with distinct sections for DLL, BLOCK RAM, CLBs, and I/O LOGIC.
FPGAs: Routing

Single-length line Switch Matrix connections

Six pass transistors per switch matrix interconnect point

Double-length lines in CLB array
FPGAs: CLB

Diagram showing a block diagram of a CLB (Configurable Logic Block) with various inputs and outputs such as G4, G3, G2, G1, F5IN, BY, SR, F4, F3, F2, F1, BX, CIN, CLK, CE, I3, I4, I2, I1, O, Look-Up Table, Carry and Control Logic, D, S, Q, CK, EC, R, X, XB, Y, YB, YQ, XQ, COUT.
PLAs/CPLDs: The 22v10

Asynchronous Reset (to all registers)
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

<table>
<thead>
<tr>
<th>31</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>eax</td>
<td>Mostly</td>
</tr>
</tbody>
</table>
| ebx| General-
| ecx| Purpose  |
| edx| Registers|
| esi| Source index|
| edi| Destination index|
| ebp| Base pointer |
| esp| Stack pointer |

<table>
<thead>
<tr>
<th>15</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs</td>
<td>Code segment</td>
</tr>
<tr>
<td>ds</td>
<td>Data segment</td>
</tr>
<tr>
<td>ss</td>
<td>Stack segment</td>
</tr>
<tr>
<td>es</td>
<td>Extra segment</td>
</tr>
<tr>
<td>fs</td>
<td>Data segment</td>
</tr>
<tr>
<td>gs</td>
<td>Data segment</td>
</tr>
</tbody>
</table>
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 %edx,%ecx
    movl -4(%ebp),%ebx
leave
ret
SPARC Programmer’s Model

### Global Registers
- r0
- r1
- r2
- r7
- r8/o0

### Output Registers
- r14/o6
- r15/o7
- r16/l0
- r23/l7

### Input Registers
- r24/i0

### Local Registers
- r16/l0

### Stack Pointer
- r14/o6

### Frame Pointer
- r30/i6
- r31/i7

### Return Address
- r31/i7

### Status Word
- PSW

### Program Counter
- PC

### Next PC
- nPC

### Program Counter
- PC
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.
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
# DSP 56000 Programmer’s Model

### Source Registers

<table>
<thead>
<tr>
<th>x1</th>
<th>x0</th>
</tr>
</thead>
<tbody>
<tr>
<td>y1</td>
<td>y0</td>
</tr>
</tbody>
</table>

### Accumulator Registers

<table>
<thead>
<tr>
<th>a2</th>
<th>a1</th>
<th>a0</th>
</tr>
</thead>
<tbody>
<tr>
<td>b2</td>
<td>b1</td>
<td>b0</td>
</tr>
</tbody>
</table>

### Address Registers

<table>
<thead>
<tr>
<th>r7</th>
<th>n7</th>
<th>m7</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>r4</td>
<td>n4</td>
<td>m4</td>
</tr>
<tr>
<td>r3</td>
<td>n3</td>
<td>m3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>r0</td>
<td>n0</td>
<td>m0</td>
</tr>
</tbody>
</table>

### Program Counter

| 15 | 0 |

### Status Register

<table>
<thead>
<tr>
<th>Program Counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Register</td>
</tr>
<tr>
<td>Loop Address</td>
</tr>
<tr>
<td>Loop Count</td>
</tr>
</tbody>
</table>

### Stack Registers

- **PC Stack**
  - 15
  - 0
- **SR Stack**
  - 15
  - 0
- **Stack pointer**
Motorola DSP56301 ALU

X Data Bus
Y Data Bus
P Data Bus

Immediate Field

Bit Field Unit
and Barrel Shifter

Accumulator
Shifter
Immediate Field

48
56
24
24
56
56
56
56

X0
X1
Y0
Y1

MUX

Multiplier

Pipeline Register

Accumulator
and Rounding Unit

A (56)
B (56)

Forwarding Register

Accumulator
Shifter

56
56
56

Shifter/Limiter

24
24
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
FIR in One ’C6 Assembly Instruction

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 in parallel
AX88796 Ethernet Controller

- MAC Core & PHY+
- Transceiver
- 8K*16 SRAM and Memory Arbiter
- Remote DMA FIFOs
- NE2000 Registers
- Host Interface
- STA
- MAC Core & PHY+
- Transceiver
- Print Port or General I/O
- SPP / GPIO
- SEEPROM I/F
- Remote DMA FIFOs
- MII I/F
- SMDC
- SMDIO
- TPI, TPO
- MM I/F
- Ctl BUS
- SA[9:0]
- SD[15:0]
<table>
<thead>
<tr>
<th>OFFSET</th>
<th>READ</th>
<th>WRITE</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>Command Register (CR)</td>
<td>Command Register (CR)</td>
</tr>
<tr>
<td>01H</td>
<td>Page Start Register (PSTART)</td>
<td>Page Start Register (PSTART)</td>
</tr>
<tr>
<td>02H</td>
<td>Page Stop Register (PSTOP)</td>
<td>Page Stop Register (PSTOP)</td>
</tr>
<tr>
<td>03H</td>
<td>Boundary Pointer (BNRY)</td>
<td>Boundary Pointer (BNRY)</td>
</tr>
<tr>
<td>04H</td>
<td>Transmit Status Register (TSR)</td>
<td>Transmit Page Start Address (TPSR)</td>
</tr>
<tr>
<td>05H</td>
<td>Number of Collisions Register (NCR)</td>
<td>Transmit Byte Count Register 0 (TBCR0)</td>
</tr>
<tr>
<td>06H</td>
<td>Current Page Register (CPR)</td>
<td>Transmit Byte Count Register 1 (TBCR1)</td>
</tr>
<tr>
<td>07H</td>
<td>Interrupt Status Register (ISR)</td>
<td>Interrupt Status Register (ISR)</td>
</tr>
<tr>
<td>08H</td>
<td>Current Remote DMA Address 0 (CRDA0)</td>
<td>Remote Start Address Register 0 (RSAR0)</td>
</tr>
<tr>
<td>09H</td>
<td>Current Remote DMA Address 1 (CRDA1)</td>
<td>Remote Start Address Register 1 (RSAR1)</td>
</tr>
<tr>
<td>0AH</td>
<td>Reserved</td>
<td>Remote Byte Count 0 (RBCR0)</td>
</tr>
<tr>
<td>0BH</td>
<td>Reserved</td>
<td>Remote Byte Count 1 (RBCR1)</td>
</tr>
<tr>
<td>0CH</td>
<td>Receive Status Register (RSR)</td>
<td>Receive Configuration Register (RCR)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REGISTER FUNCTION</td>
<td>SUB ADDR (HEX)</td>
<td>D7</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>----------------</td>
<td>----</td>
</tr>
<tr>
<td>Chip version: register 00H</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chip version (read only)</td>
<td>00</td>
<td>ID07</td>
</tr>
<tr>
<td>Video decoder: registers 01H to 2FH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRONTEND PART: REGISTERS 01H TO 05H</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horizontal increment control</td>
<td>01</td>
<td></td>
</tr>
<tr>
<td>Analog input control 1</td>
<td>02</td>
<td>FUSE1</td>
</tr>
<tr>
<td>Analog input control 2</td>
<td>03</td>
<td>(i)</td>
</tr>
<tr>
<td>Analog input control 3</td>
<td>04</td>
<td>GA117</td>
</tr>
<tr>
<td>Analog input control 4</td>
<td>05</td>
<td>GA127</td>
</tr>
<tr>
<td>DECODER PART: REGISTERS 06H TO 2FH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horizontal sync start</td>
<td>06</td>
<td>HSB7</td>
</tr>
<tr>
<td>Horizontal sync stop</td>
<td>07</td>
<td>HSS7</td>
</tr>
<tr>
<td>Sync control</td>
<td>08</td>
<td>AUFQ</td>
</tr>
<tr>
<td>Luminance control</td>
<td>09</td>
<td>BYPS</td>
</tr>
<tr>
<td>Luminance brightness control</td>
<td>0A</td>
<td>DBRI7</td>
</tr>
<tr>
<td>Luminance contrast control</td>
<td>0B</td>
<td>DCON7</td>
</tr>
<tr>
<td>Chrominance saturation control</td>
<td>0C</td>
<td>DSAT7</td>
</tr>
<tr>
<td>Chrominance hue control</td>
<td>0D</td>
<td>HUEC7</td>
</tr>
<tr>
<td>Chrominance control 1</td>
<td>0E</td>
<td>CDTQ</td>
</tr>
<tr>
<td>Chrominance gain control 1</td>
<td>0F</td>
<td>AGCQ</td>
</tr>
<tr>
<td>Chrominance control 2</td>
<td>10</td>
<td>OFFQ</td>
</tr>
<tr>
<td>Mode/delay control</td>
<td>11</td>
<td>COLO</td>
</tr>
<tr>
<td>RT signal control</td>
<td>12</td>
<td>RTSE13</td>
</tr>
<tr>
<td>RT/Port output control</td>
<td>13</td>
<td>RTCE</td>
</tr>
<tr>
<td>Analog/ADC/compatibility control</td>
<td>14</td>
<td>CM99</td>
</tr>
<tr>
<td>VGATE start, FID change</td>
<td>15</td>
<td>VSTA7</td>
</tr>
<tr>
<td>VGATE stop</td>
<td>16</td>
<td>VST07</td>
</tr>
<tr>
<td>Miscellaneous/VGATE MSBs</td>
<td>17</td>
<td>LLCE</td>
</tr>
</tbody>
</table>
Fixed-function: The 7400 series

7400
Quad NAND Gate

74374
Octal D Flip-Flop