Processors, FPGAs, and ASICs

Stephen A. Edwards

Columbia University

Spring 2013
Spectrum of IC choices

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

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

Ammeter

0 V

3 V
An N-Channel MOS Transistor

Gate positive: On

SiO$_2$

Drain

Source

p (holes)

Ammeter
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
Standard Cell ASICs
Standard Cell ASICs
Channeled Gate Arrays
Sea-of-Gates Gate Arrays
FPGAs: Floorplan
FPGAs: Routing

Single-length line Switch Matrix connections

Six pass transistors per switch matrix interconnect point

Double-length lines in CLB array

Switch matrices
PLAs/CPLDs: The 22v10

Asynchronous Reset (to all registers)

First Fuse Numbers

Macro-cell
R = 5809
P = 5808
R = 5811
P = 5810
R = 5813
P = 5812
R = 5815
P = 5814
R = 5817
P = 5816

I/O/Q

CLK/I
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>
<tr>
<td>ebx</td>
<td>General-Purpose</td>
</tr>
<tr>
<td>ecx</td>
<td>Registers</td>
</tr>
<tr>
<td>edx</td>
<td></td>
</tr>
<tr>
<td>esi</td>
<td>Source index</td>
</tr>
<tr>
<td>edi</td>
<td>Destination index</td>
</tr>
<tr>
<td>ebp</td>
<td>Base pointer</td>
</tr>
<tr>
<td>esp</td>
<td>Stack pointer</td>
</tr>
<tr>
<td>eflags</td>
<td>Status word</td>
</tr>
<tr>
<td>eip</td>
<td>Instruction Pointer</td>
</tr>
</tbody>
</table>

<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 -4(%ebp),%ebx
leave
ret
### SPARC Programmer’s Model

<table>
<thead>
<tr>
<th>31 0</th>
<th>Always 0</th>
<th>31 0</th>
<th>Input Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>r0</td>
<td>Global Registers</td>
<td>r24/i0</td>
<td>Frame Pointer</td>
</tr>
<tr>
<td>r1</td>
<td></td>
<td>:</td>
<td>Return Address</td>
</tr>
<tr>
<td>:</td>
<td>Output Registers</td>
<td>:</td>
<td></td>
</tr>
<tr>
<td>r7</td>
<td></td>
<td>r30/i6</td>
<td></td>
</tr>
<tr>
<td>r8/o0</td>
<td></td>
<td>r31/i7</td>
<td></td>
</tr>
<tr>
<td>:</td>
<td>Stack Pointer</td>
<td>:</td>
<td>Status Word</td>
</tr>
<tr>
<td>r14/o6</td>
<td></td>
<td>PSW</td>
<td>Program Counter</td>
</tr>
<tr>
<td>r15/o7</td>
<td></td>
<td>PC</td>
<td>Next PC</td>
</tr>
<tr>
<td>r16/l0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>:</td>
<td>Local Registers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r23/l7</td>
<td></td>
<td>nPC</td>
<td></td>
</tr>
</tbody>
</table>
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
Motorola DSP56301

Figure 1-1.
PLL OnCE™
Clock Generator
Internal
Data Bus
Switch
Program RAM
4096 × 24 (Default)
YAB
XAB
PA...
Power Management
External Bus Interface and I-Cache Control
External Address Bus Switch
5 DE
MAC

Memory Expansion Area

Peripheral Expansion Area

24-Bit DSP56300 Core

Data ALU
24 × 24 + 56 → 56-bit MAC
Two 56-bit Accumulators
56-bit Barrel Shifter

Address Generation Unit
Six-Channel DMA Unit

Program Interrupt Controller
Program Decode Controller
Program Address Generator

Clock Generator
PLL

External Bus Interface

Address
CONTROL
DATA
DE

Program RAM

X Data RAM
2048 × 24 (Default)
DDB
YAB
XAB
PAB
DAB

Y Data RAM
2048 × 24 (Default)
YDB
XDB
PDB

GDB

External Data Bus Switch

Power Management

RESET
PINIT/NMI

EXTAL
XTAL

Modem Control

Boot-strap ROM
Internal Data Bus Switch

JTAG
OnCE™
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

<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>r4</td>
<td>n4</td>
<td>m4</td>
</tr>
<tr>
<td>r3</td>
<td>n3</td>
<td>m3</td>
</tr>
<tr>
<td>r0</td>
<td>n0</td>
<td>m0</td>
</tr>
</tbody>
</table>

Program Counter

| 15 | 0 |

Status Register

| 15 | 0 |

Loop Address

| 15 | 0 |

Loop Count

| 15 | 0 |

PC Stack

| 15 | 0 |

SR Stack

| 15 | 0 |

Stack pointer

|     |
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

Data path A

Data path B

Register file A (A0–A15)

Register file B (B0–B15)

Control register file
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

- 8K* 16 SRAM and Memory Arbiter
- SEEPROM I/F
- SPP / GPIO
- NE2000 Registers
- Remote DMA FIFOs
- Host Interface
- STA
- MAC Core & PHY+
- Tranceiver
- Ctl BUS
- SA[9:0]
- SD[15:0]
- EECS
- EECK
- EEDI
- EEDO
- Print Port or General I/O
- TPI, TPO
- MII I/F
- SMDC
- SMDIO
## Ethernet Controller Registers

**PAGE 0 (PS1=0,PS0=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>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>FNTEND PART: REGISTERS 01H TO 05H</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horizontal increment delay</td>
<td>01</td>
<td>(1)</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>(1)</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>AUFD</td>
</tr>
<tr>
<td>Luminance control</td>
<td>09</td>
<td>BYPS</td>
</tr>
<tr>
<td>Luminance brightness control</td>
<td>0A</td>
<td>DBR17</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>CDTO</td>
</tr>
<tr>
<td>Chrominance gain control 1</td>
<td>0F</td>
<td>ACWG</td>
</tr>
<tr>
<td>Chrominance gain control 2</td>
<td>10</td>
<td>OFFU1</td>
</tr>
<tr>
<td>Mode/delay control</td>
<td>11</td>
<td>COLO</td>
</tr>
<tr>
<td>FT signal control</td>
<td>12</td>
<td>RTSE13</td>
</tr>
<tr>
<td>FT/X-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