Processors, FPGAs, and ASICs
Part 2: Processors to Fixed-Function

Stephen A. Edwards

Columbia University

Spring 2024
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., Accelerometer)
- Part number (e.g., 7400)

Cheap, quick to design
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>Register</th>
<th>Usage</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>eax</td>
<td>Mostly General-Purpose Registers</td>
<td>cs</td>
</tr>
<tr>
<td>ebx</td>
<td></td>
<td>ds</td>
</tr>
<tr>
<td>ecx</td>
<td></td>
<td>ss</td>
</tr>
<tr>
<td>edx</td>
<td></td>
<td>es</td>
</tr>
<tr>
<td>esi</td>
<td>Source index</td>
<td>fs</td>
</tr>
<tr>
<td>edi</td>
<td>Destination index</td>
<td>gs</td>
</tr>
<tr>
<td>ebp</td>
<td>Base pointer</td>
<td></td>
</tr>
<tr>
<td>esp</td>
<td>Stack pointer</td>
<td></td>
</tr>
<tr>
<td>eflags</td>
<td>Status word</td>
<td></td>
</tr>
<tr>
<td>eip</td>
<td>Instruction Pointer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
int gcd(int m, int n)
{
    int r;
    while ((r = m % n) != 0) {
        m = n;
        n = r;
    }
    return n;
}
Sun's SPARC Processor c. 1987
**SPARC Programmer’s Model**

<table>
<thead>
<tr>
<th>31</th>
<th>0</th>
<th>Always 0</th>
<th>31</th>
<th>0</th>
<th>Local Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>r0</td>
<td>r16/l0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r1</td>
<td></td>
<td>r17/l1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r8/o0</td>
<td>r23/l7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>r24/i0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r14/o6</td>
<td>r30/i6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r15/o7</td>
<td>r31/i7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stack Pointer</td>
<td>Frame Pointer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Global Registers

Output Registers

Input Registers

Frame Pointer

Return Address

Status Register

Program Counter

Next 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.
int gcd(m, n)
int m, n;
{
    int r;
    while ((r = m % n) != 0) {
        m = n;
        n = r;
    }
    return n;
}
Motorola’s DSP56000 c. 1986
DSP 56000 Programmer’s Model

<table>
<thead>
<tr>
<th>55 48 47</th>
<th>24 23</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>x0</td>
<td></td>
</tr>
<tr>
<td>y1</td>
<td>y0</td>
<td></td>
</tr>
</tbody>
</table>

| a2 | a1 | a0 |
| b2 | b1 | b0 |

Source Registers

<table>
<thead>
<tr>
<th>15</th>
<th>0</th>
</tr>
</thead>
</table>

Accumulator Registers

<table>
<thead>
<tr>
<th>15</th>
<th>0</th>
<th>15</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>r7</td>
<td>n7</td>
<td>m7</td>
<td></td>
</tr>
</tbody>
</table>

Address Registers

<table>
<thead>
<tr>
<th>15</th>
<th>0</th>
<th>15</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>r7</td>
<td>n7</td>
<td>m7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Program Counter

Status Register

Loop Address

Loop Count

PC Stack

SR Stack

Stack pointer
Motorola DSP56000
Data ALU
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
TMS320
C6201
VLIW
DSP
c. 1997
FIR in One ’C6 Assembly Instruction

FIRLOOP:

LDH .D1  *A1++, A2 ; Fetch next sample
||    LDH .D2  *B1++, B2 ; Fetch next coefficient
|| [B0]    SUB .L2  B0, 1, B0 ; Decrement loop count
|| [B0]  B  .S2   FIRLOOP ; Branch if non-zero
||    MPY .M1X A2, B2, A3 ; Sample × Coefficient
||    ADD .L1 A4, A3, A4 ; Accumulate result
FIR in One ’C6 Assembly Instruction

FIRLOOP:

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

Run in parallel
FIR in One ’C6 Assembly Instruction

Load a halfword (16 bits)

FIRLOOP:

LDH .D1 *A1++, A2 ; Fetch next sample

LDH .D2 *B1++, B2 ; Fetch next coefficient

[B0] SUB .L2 B0, 1, B0 ; Decrement loop count

[B0] B .S2 FIRLOOP ; Branch if non-zero

MPY .M1X A2, B2, A3 ; Sample × Coefficient

ADD .L1 A4, A3, A4 ; Accumulate result
FIR in One ’C6 Assembly Instruction

FIRLOOP:

- LDH .D1 *A1++, A2 ; Fetch next sample
- LDH .D2 *B1++, B2 ; Fetch next coefficient
- [B0] SUB .L2 B0, 1, B0 ; Decrement loop count
- [B0] B .S2 FIRLOOP ; Branch if non-zero
- MPY .M1X A2, B2, A3 ; Sample × Coefficient
- ADD .L1 A4, A3, A4 ; Accumulate result
FIR in One ’C6 Assembly Instruction

FIRLOOP:

LDH .D1 *A1++, A2 ; Fetch next sample

LDH .D2 *B1++, B2 ; Fetch next coefficient

[B0] SUB .L2 B0, 1, B0 ; Decrement loop count

[B0] B .S2 FIRLOOP ; Branch if non-zero

MPY .M1X A2, B2, A3 ; Sample × Coefficient

ADD .L1 A4, A3, A4 ; Accumulate result

Use the cross path
FIR in One ‘C6 Assembly Instruction

FIRLOOP:

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

Predicated instruction (only if B0 non-zero)
Analog Devices ADXL345 Accelerometer

3-AXIS SENSOR
SENSE ELECTRONICS
DIGITAL FILTER
32 LEVEL FIFO
CONTROL AND INTERRUPT LOGIC
SERIAL I/O
POWER MANAGEMENT

- VS
- VDD I/O
- INT1
- INT2
- SDA/SDI/SDIO
- SDO/ALT ADDRESS
- SCL/SCLK
- GND
- CS
14 pins, 3mm by 5mm
DE1-SoC Connections to the ADXL345 Accelerometer

```
HPS_I2C1_SCLK  HPS_I2C1_SDAT  HPS_GSENSOR_INT
              \             \             \            \_______
               |             |             |            |       
               |             |             |            |       
               |             |             |            |       
               \             \             \            \_______
                SCL_SCLK    SDA_SDI_SDIO  INT1
                                      \___
                                        ADXL345
```
I²C Bus Protocol

![Diagram of I²C Bus Protocol](image-url)
<table>
<thead>
<tr>
<th>Address</th>
<th>Dec</th>
<th>Name</th>
<th>Type</th>
<th>Reset Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>0</td>
<td>DEVID</td>
<td>R</td>
<td>11100101</td>
<td>Device ID</td>
</tr>
<tr>
<td>0x01 to 0x1C</td>
<td>1 to 28</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>Reserved; do not access</td>
</tr>
<tr>
<td>0x1D</td>
<td>29</td>
<td>THRESH_TAP</td>
<td>R/W</td>
<td>00000000</td>
<td>Tap threshold</td>
</tr>
<tr>
<td>0x1E</td>
<td>30</td>
<td>OFSX</td>
<td>R/W</td>
<td>00000000</td>
<td>X-axis offset</td>
</tr>
<tr>
<td>0x1F</td>
<td>31</td>
<td>OFSY</td>
<td>R/W</td>
<td>00000000</td>
<td>Y-axis offset</td>
</tr>
<tr>
<td>0x20</td>
<td>32</td>
<td>OFSZ</td>
<td>R/W</td>
<td>00000000</td>
<td>Z-axis offset</td>
</tr>
<tr>
<td>0x21</td>
<td>33</td>
<td>DUR</td>
<td>R/W</td>
<td>00000000</td>
<td>Tap duration</td>
</tr>
<tr>
<td>0x22</td>
<td>34</td>
<td>Latent</td>
<td>R/W</td>
<td>00000000</td>
<td>Tap latency</td>
</tr>
<tr>
<td>0x23</td>
<td>35</td>
<td>Window</td>
<td>R/W</td>
<td>00000000</td>
<td>Tap window</td>
</tr>
<tr>
<td>0x24</td>
<td>36</td>
<td>THRESH_ACT</td>
<td>R/W</td>
<td>00000000</td>
<td>Activity threshold</td>
</tr>
<tr>
<td>0x25</td>
<td>37</td>
<td>THRESH_INACT</td>
<td>R/W</td>
<td>00000000</td>
<td>Inactivity threshold</td>
</tr>
<tr>
<td>0x26</td>
<td>38</td>
<td>TIME_INACT</td>
<td>R/W</td>
<td>00000000</td>
<td>Inactivity time</td>
</tr>
<tr>
<td>0x27</td>
<td>39</td>
<td>ACT_INACT_CTL</td>
<td>R/W</td>
<td>00000000</td>
<td>Axis enable control for activity and inactivity detection</td>
</tr>
<tr>
<td>0x28</td>
<td>40</td>
<td>THRESH_FF</td>
<td>R/W</td>
<td>00000000</td>
<td>Free-fall threshold</td>
</tr>
<tr>
<td>0x29</td>
<td>41</td>
<td>TIME_FF</td>
<td>R/W</td>
<td>00000000</td>
<td>Free-fall time</td>
</tr>
<tr>
<td>0x2A</td>
<td>42</td>
<td>TAP_AXES</td>
<td>R/W</td>
<td>00000000</td>
<td>Axis control for single tap/double tap</td>
</tr>
<tr>
<td>0x2B</td>
<td>43</td>
<td>ACT_TAP_STATUS</td>
<td>R</td>
<td>00000000</td>
<td>Source of single tap/double tap</td>
</tr>
<tr>
<td>0x2C</td>
<td>44</td>
<td>BW_RATE</td>
<td>R/W</td>
<td>00001010</td>
<td>Data rate and power mode control</td>
</tr>
<tr>
<td>0x2D</td>
<td>45</td>
<td>POWER_CTL</td>
<td>R/W</td>
<td>00000000</td>
<td>Power-saving features control</td>
</tr>
<tr>
<td>0x2E</td>
<td>46</td>
<td>INT_ENABLE</td>
<td>R/W</td>
<td>00000000</td>
<td>Interrupt enable control</td>
</tr>
<tr>
<td>0x2F</td>
<td>47</td>
<td>INT_MAP</td>
<td>R/W</td>
<td>00000000</td>
<td>Interrupt mapping control</td>
</tr>
<tr>
<td>0x30</td>
<td>48</td>
<td>INT_SOURCE</td>
<td>R</td>
<td>00000010</td>
<td>Source of interrupts</td>
</tr>
<tr>
<td>0x31</td>
<td>49</td>
<td>DATA_FORMAT</td>
<td>R/W</td>
<td>00000000</td>
<td>Data format control</td>
</tr>
<tr>
<td>0x32</td>
<td>50</td>
<td>DATAX0</td>
<td>R</td>
<td>00000000</td>
<td>X-Axis Data 0</td>
</tr>
<tr>
<td>0x33</td>
<td>51</td>
<td>DATAX1</td>
<td>R</td>
<td>00000000</td>
<td>X-Axis Data 1</td>
</tr>
<tr>
<td>0x34</td>
<td>52</td>
<td>DATAY0</td>
<td>R</td>
<td>00000000</td>
<td>Y-Axis Data 0</td>
</tr>
<tr>
<td>0x35</td>
<td>53</td>
<td>DATAY1</td>
<td>R</td>
<td>00000000</td>
<td>Y-Axis Data 1</td>
</tr>
<tr>
<td>0x36</td>
<td>54</td>
<td>DATAZ0</td>
<td>R</td>
<td>00000000</td>
<td>Z-Axis Data 0</td>
</tr>
<tr>
<td>0x37</td>
<td>55</td>
<td>DATAZ1</td>
<td>R</td>
<td>00000000</td>
<td>Z-Axis Data 1</td>
</tr>
<tr>
<td>0x38</td>
<td>56</td>
<td>FIFO_CTL</td>
<td>R/W</td>
<td>00000000</td>
<td>FIFO control</td>
</tr>
<tr>
<td>0x39</td>
<td>57</td>
<td>FIFO_STATUS</td>
<td>R</td>
<td>00000000</td>
<td>FIFO status</td>
</tr>
</tbody>
</table>
REGISTER DEFINITIONS
Register 0x00—DEVID (Read Only)
The DEVID register is eight bits and includes the threshold value for detecting inactivity. The data format is unsign., so the magnitude of the inactivity event is compared with the value in the THRESH_INACT register. The scale factor is 62.5 mLSB. A value of 0 may result in unreadable behavior if single tap/double tap interrupts are enabled.

Register 0x16—Register 0x17—ACDC and INACT/ACDC Bits

A setting of 0 selects dc-coupled operation, and a setting of 1 enables ac-coupled operation. In dc-coupled operation, the current acceleration magnitude is compared directly with THRESH_INACT and THRESH_ACT to determine whether activity or inactivity is occurring. In ac-coupled operation for activity detection, the acceleration value at the start of activity detection is taken as a reference value. New samples of acceleration are then compared to this reference value, and if the magnitude of the difference exceeds the THRESH_INACT value and the current acceleration is in the THRESH_INACT—the inactivity function uses filtered output data representing a sleep mode.

Register 0x18—TAP_ENABLE (Read/Write)

This register is used to enable single tap/double tap interrupts. A value of 1 enables the corresponding tap function.

Register 0x19—TAP_CTL (Read/Write)

This register is used to control the single tap/double tap function. A setting of 0 enables one tap after the current tap/double tap event occurs. A value of 1 enables a second tap after the current tap/double tap event occurs. A setting of 0 enables a second tap after the current tap/double tap event occurs. A value of 1 enables a second tap after the current tap/double tap event occurs.

Adapt Bit
A setting of 1 in the adapt bit indicates that the part is asleep. A setting of 1 indicates that the part is asleep. The bit toggles only if the device is configured for auto sleep. See the AUTO_SLEEP Bit section for more information on auto sleep mode.

Register 0x22—INTERRUPT (Read/Write)
The INTERRUPT register is eight bits and contains the threshold value for detecting activity. A setting of 0 disables the selected input; if an input is enabled, the function is disabled. For activity detection, all participating axes are logically ORed, causing the activity function to trigger when any of the participating axes exceeds the threshold. For inactivity detection, all participating axes are logically ANDed, causing the inactivity function to trigger only if all participating axes are below the threshold for the specified time.

Register 0x23—Window (Read/Write)

The Window register is eight bits and contains an unsign. time value representing the time interval that an event must occur above the THRESH_ACT threshold to qualify as an event. The scale factor is 62.5 mLSB. A value of 0 disables the single tap/double tap function.

Register 0x24—TAP﻿# (Read/Write)
The TAP﻿# register is eight bits and contains the threshold value for detecting activity. A setting of 0 disables the corresponding tap function. If an input is enabled, the function is disabled. For activity detection, all participating axes are logically ORed, causing the activity function to trigger when any of the participating axes exceeds the threshold. For inactivity detection, all participating axes are logically ANDed, causing the inactivity function to trigger only if all participating axes are below the threshold for the specified time.

Register 0x25—MEM (Read/Write)

The MEM register is eight bits and contains an unsign. time value representing the time interval that an event must occur above the THRESH_ACT threshold to qualify as an event. The scale factor is 62.5 mLSB. A value of 0 disables the single tap/double tap function.

Register 0x26—TIME_INACT (Read/Write)
The TIME_INACT register is eight bits and contains the threshold value for detecting activity. A setting of 0 disables the corresponding tap function. If an input is enabled, the function is disabled. For activity detection, all participating axes are logically ORed, causing the activity function to trigger when any of the participating axes exceeds the threshold. For inactivity detection, all participating axes are logically ANDed, causing the inactivity function to trigger only if all participating axes are below the threshold for the specified time.

Register 0x27—ACT_INACT_CTL (Read/Write)

This register is used to control the single tap/double tap function. A setting of 0 enables one tap after the current tap/double tap event occurs. A value of 1 enables a second tap after the current tap/double tap event occurs. A setting of 0 enables a second tap after the current tap/double tap event occurs. A value of 1 enables a second tap after the current tap/double tap event occurs.

Register 0x30—INT_SOURCE (Read Only)
The INT_SOURCE register is eight bits and contains the threshold value for detecting activity. A setting of 0 disables the selected input; if an input is enabled, the function is disabled. For activity detection, all participating axes are logically ORed, causing the activity function to trigger when any of the participating axes exceeds the threshold. For inactivity detection, all participating axes are logically ANDed, causing the inactivity function to trigger only if all participating axes are below the threshold for the specified time.

Register 0x33—FIFO_CTL (Read/Write)
The FIFO_CTL register is eight bits and contains the threshold value for detecting activity. A setting of 0 disables the selected input; if an input is enabled, the function is disabled. For activity detection, all participating axes are logically ORed, causing the activity function to trigger when any of the participating axes exceeds the threshold. For inactivity detection, all participating axes are logically ANDed, causing the inactivity function to trigger only if all participating axes are below the threshold for the specified time.
Analog Devices ADV7180 Video Decoder

- **ANALOG VIDEO INPUTS**: AIN1, AIN2, AIN3, AIN4, AIN5, AIN6
- **PLL**
- **ADLLT PROCESSING**
- **CLOCK PROCESSING BLOCK**
- **10-BIT, 86MHz ADC**
- **2D COMB**
- **VBI SLICER**
- **COLOR DEMOD**
- **DIGITAL PROCESSING BLOCK**
- **FIFO**
- **OUTPUT BLOCK**
- **ADV7180 SHA A/D**
- **VS**
- **FIELD2**
- **GPO1**
- **SFL**
- **INTRQ**
- **SCLK, SDATA, ALSB, RESET, PWRDWN**

**DIAGRAM:**
- XTAL1 connects to XTAL
- ANALOG VIDEO INPUTS are connected to MUX BLOCK
- PLL connects to ADLLT PROCESSING and C clk
- 10-BIT, 86MHz ADC connects to PLL and SHA
- SHA connects to A/D and Reference
- Digital Processing Block connects to FIFO
- FIFO connects to Output Block
- Output Block connects to ADV7180 SHA A/D
- ADV7180 SHA A/D connects to Digital Processing Block
- Analog Devices ADV7180 Video Decoder
<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
<th>Register Name</th>
<th>RW</th>
<th>Value (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>VID_SEL[3]</td>
<td>RW</td>
<td>0110</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>VID_SEL[2]</td>
<td>RW</td>
<td>0001</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>VID_SEL[1]</td>
<td>RW</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>VID_SEL[0]</td>
<td>RW</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CB[6]</td>
<td>SD</td>
<td>0001</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CB[5]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CB[4]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CB[3]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CB[2]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CR[4]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CR[3]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CR[2]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CR[1]</td>
<td>SD</td>
<td>0000</td>
</tr>
<tr>
<td>0x007F</td>
<td>Input Control</td>
<td>SD_SAT_CR[0]</td>
<td>SD</td>
<td>0000</td>
</tr>
</tbody>
</table>

Note: The table represents the hexadecimal values for various registers and bits within those registers, with columns for address, description, register name, read/write (RW) permissions, and hexadecimal value. The table is organized in a standard format of rows and columns, with each row corresponding to a unique address and its associated register details.
Fixed-function: The 7400 series

7400
Quad NAND Gate

74374
Octal D Flip-Flop
The 74181 4-bit ALU
The 74181 4-bit ALU