Pipelining

- sequential
- pipelined
- superscalar
- superpipelined

Typical DSP:
- Fetch, Decode, Read, Execute

Managing pipelines:
- compiler or programmer
- interlocking
- hardware instruction scheduling
Pipeline operation and programming style

- **Time-stationary** pipeline model
  - Programmer controls each cycle
  - Motorola DSP56001:
    - $\text{MAC } X_0, Y_0, A \quad X: (R0) +, X_0 \quad Y: (R4) -, Y_0$

- **Data-stationary** pipeline model
  - Programmer specifies data operations
  - TMS320C30/40
    - $\text{mpyf } *++\text{ar0}(l), *++\text{ar1}(\text{ir0}), r0$

- **Interlocking** pipeline
  - Programmer is “protected” from pipeline effects

---

Pipeline hazards

- A **control** hazard occurs when a branch instruction is decoded
  - “Flush” the pipeline
  - or: Delayed branch (expose pipeline)
- A **data** hazard occurs because an operand cannot be read yet
  - Assume programmer did it deliberately
  - or: Interlock hardware inserts “bubble”
Avoiding control hazards: hardware looping

- A repeat instruction repeats one or a block of following instructions
- The pipeline is filled with the repeated instruction (or block of instructions)
- Cost: one pipeline flush only

A key factor in the numeric performance of DSPs is the provision of special hardware to perform looping.

RPT COUNT
TBLR **+

Each tap requires
- fetching one data sample
- fetching one operand
- multiplying
- accumulating
- shifting one sample in the delay line

Implies of the architecture:
- at least 3 memory accesses per instruction
- auto-increment or decrement addressing modes
**DSP-specific addressing features**

- **modulo addressing**
  - implements circular buffers and delay lines

- **bit reversed addressing**
  - used to implement the radix-2 FFT

---

**Data-shifting**

<table>
<thead>
<tr>
<th>Time</th>
<th>Buffer contents</th>
<th>Next sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=N</td>
<td>(x_{N-k+1} x_{N-k+2} \ldots x_N x_{N-1} x_{N+1})</td>
<td>(\ldots x_{N+1})</td>
</tr>
<tr>
<td>n=N+1</td>
<td>(x_{N-k+2} x_{N-k+3} \ldots x_N x_{N-1} x_{N+1})</td>
<td>(\ldots x_{N+2})</td>
</tr>
<tr>
<td>n=N+2</td>
<td>(x_{N-k+3} x_{N-k+4} \ldots x_N x_{N-1} x_{N+2})</td>
<td>(\ldots x_{N+3})</td>
</tr>
</tbody>
</table>

**Modulo addressing**

<table>
<thead>
<tr>
<th>Time</th>
<th>Buffer contents</th>
<th>Next sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=N</td>
<td>(x_{N-k+1} x_{N-k+2} \ldots x_N x_{N-1} x_{N+1})</td>
<td>(\ldots x_{N+1})</td>
</tr>
<tr>
<td>n=N+1</td>
<td>(x_{N-k+2} x_{N-k+3} \ldots x_N x_{N+1} x_{N+2})</td>
<td>(\ldots x_{N+2})</td>
</tr>
<tr>
<td>n=N+2</td>
<td>(x_{N-k+3} x_{N-k+4} \ldots x_N x_{N+2} x_{N+3})</td>
<td>(\ldots x_{N+3})</td>
</tr>
</tbody>
</table>

---

**Simplified TMS320C50 Architecture**

![Diagram of TMS320C50 Architecture](image-url)
FIR Filter in TMS320C50

Memory locations:

- **COEFFP** .set 02000h ; Program mem address
- **X** .set 037Fh ; Newest data sample
- **LASTAP** .set 037FH ; Oldest data sample

```
... 
  LAR AR3,#LASTAP ; Point to oldest sample
  RPT #127
  MACD COEFFP,*- ; Do the thang
  APAC
  SACH Y,1 ; Store result -- note shift
```

Single-cycle loop

Clean-up

Initialize
DSP Cores

ASIC with:
- Programmable DSP
- RAM
- ROM
- Standard Cells
- Codec
- Peripherals
- Gate array
- Microcontroller

TMS320 cDSP (Customizable DSP) from Texas Instruments

Graphic courtesy of Texas Instruments, Inc.