

# **DSP Processors – Lecture 14**

## **Fundamentals**

**Ingrid Verbauwhede**

**Department of Electrical Engineering  
University of California Los Angeles**

ingrid@ee.ucla.edu

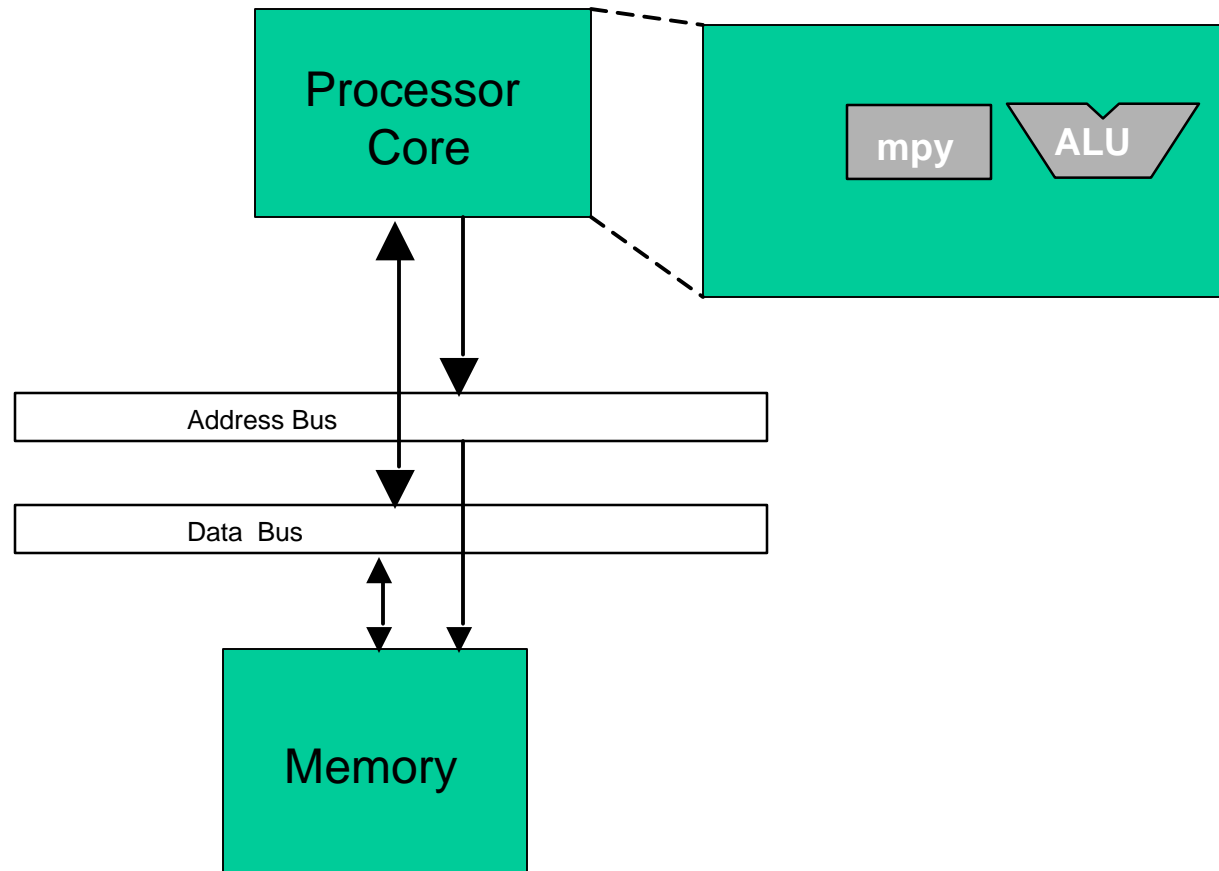
# References

- The origins:
  - E.A. Lee, “Programmable DSP Processors,” Part I, IEEE ASSP magazine, October 1988, pg. 4-19.
  - Part II, IEEE ASSP magazine, January 1989, pg. 4-14
- Good overview:
  - P. Lapsley, J. Bier, A. Shoham, E.A.Lee, “DSP Processor Fundamentals: Architectures and Features,” IEEE Press, 1998.



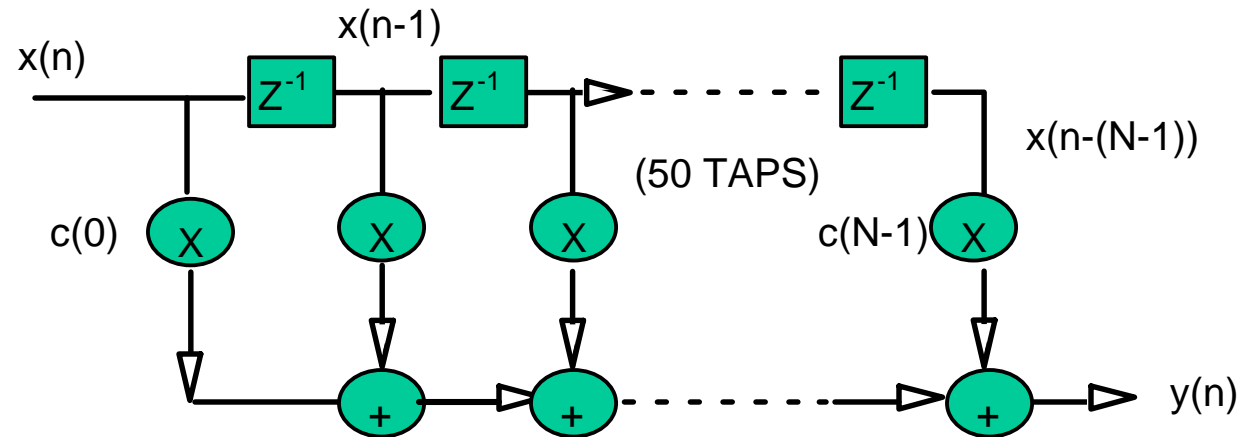
# Von Neumann machine

- One memory space



# FIR implementation

$$y(n) = \sum_{i=0}^{N-1} c(i) x(n-i)$$



$$\begin{aligned}
 y(0) &= c(0)x(0) + c(1)x(-1) + c(2)x(-2) + \dots + c(N-1)x(1-N); \\
 y(1) &= c(0)x(1) + c(1)x(0) + c(2)x(-1) + \dots + c(N-1)x(2-N); \\
 y(2) &= c(0)x(2) + c(1)x(1) + c(2)x(0) + \dots + c(N-1)x(3-N); \\
 &\dots \\
 y(n) &= c(0)x(n) + c(1)x(n-1) + c(2)x(n-2) + \dots + c(N-1)x(n-(N-1));
 \end{aligned}$$

Execute row by row

# FIR on Von Neumann

Assume Von Neumann has multiply and accumulate instruction  
(not necessarily the case)

Assume also that pipelining allows to execute the multiply and accumulate  
in parallel with the read or write operations.

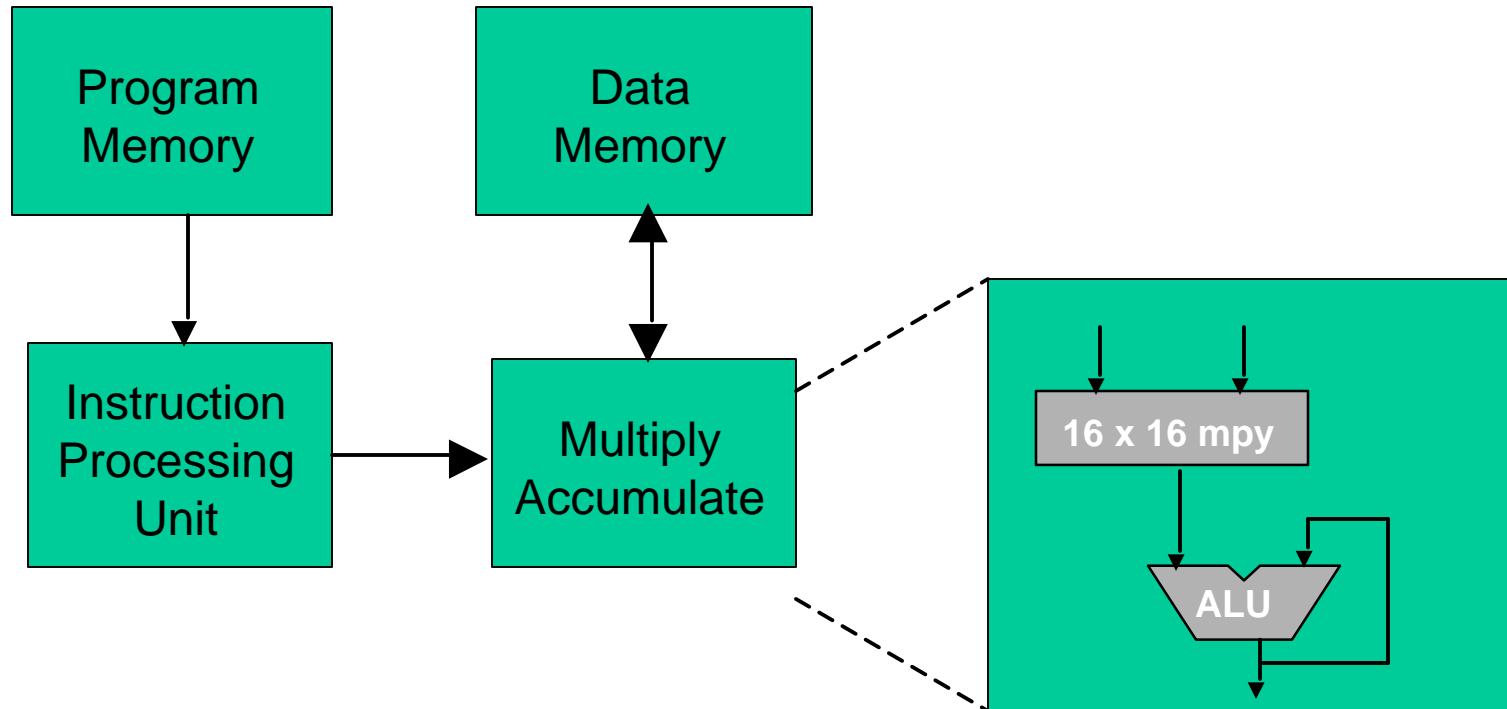
Then one tap needs 4 cycles:

1. read multiply-accumulate instruction
2. read data value from memory
3. read coefficient from memory
4. write data value to the next location in the delay line  
(because for the next sample, all values are shifted by one location)

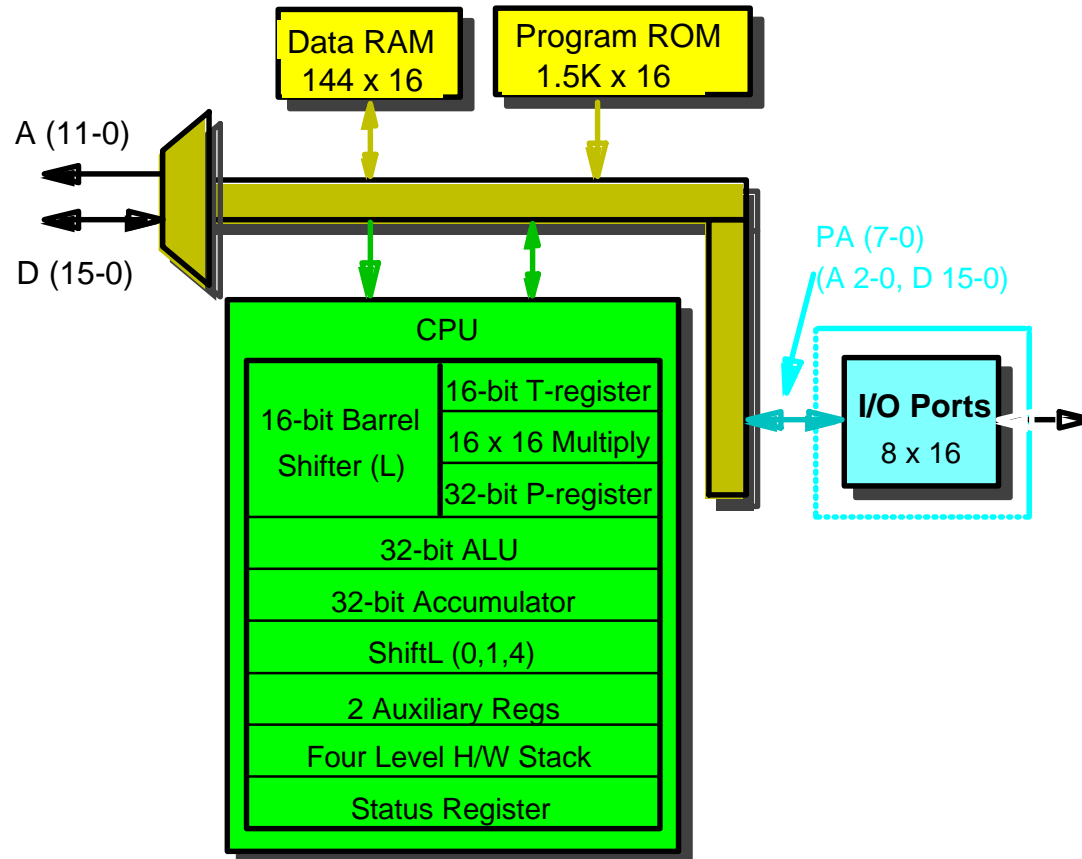
***Memory bandwidth is crucial !!!***

# Basic Harvard Architecture

- Separate data memory from program memory!



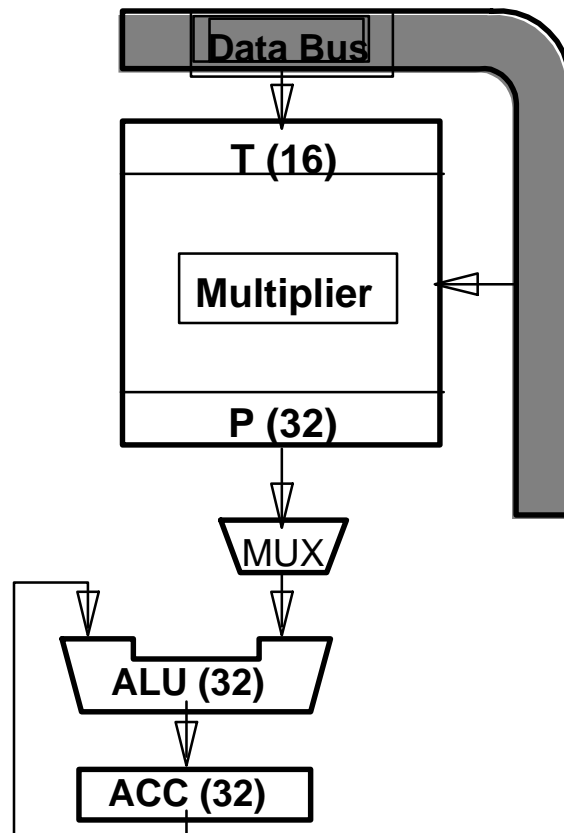
# Example 1: TMS320C10 (1982)



- **160/200ns Instruction cycle time**
- 4K word external address reach
- **60 general purpose and DSP specific instructions**
- **Single cycle multiply**
- 16-bit Barrel Shifter
- External interrupt and polled input pins
- Eight 16-bit I/O ports
- 40-pin DIP/44-pin PLCC

Courtesy: Texas Instruments

# TMS320C1x Example - Sum of Products

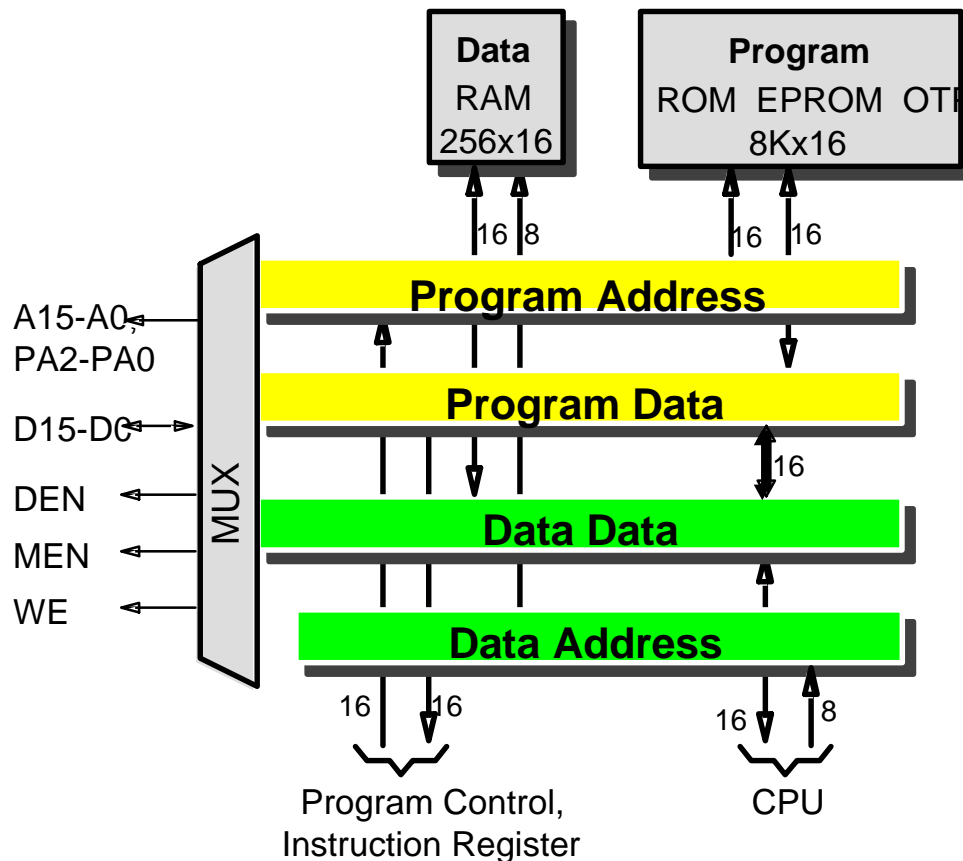


Compute  $Y = AX1 + BX2 + CX3 + DX4$

<b>ZAC</b>		ACC=0
<b>LT</b>	<b>X1</b>	T=X1
<b>MPY</b>	<b>A</b>	P=AX1
<b>LTA</b>	<b>X2</b>	ACC=AX1;T=X2
<b>MPY</b>	<b>B</b>	P=BX2
<b>LTA</b>	<b>X3</b>	ACC=AX1+BX2;T=X3
<b>MPY</b>	<b>C</b>	P=CX3
<b>LTA</b>	<b>X4</b>	ACC=AX1+BX2+CX3;T=X4
<b>MPY</b>	<b>D</b>	P=DX4
<b>APAC</b>		ACC=AX1+BX2+Cx3+DX4
<b>SACH</b>	<b>Y1</b>	STORE 32-BIT RESULT
<b>SACH</b>	<b>Y2</b>	AT LOCATIONS Y1, Y2

- 50 taps = 103 cycles
- = Program ROM of 103 instructions

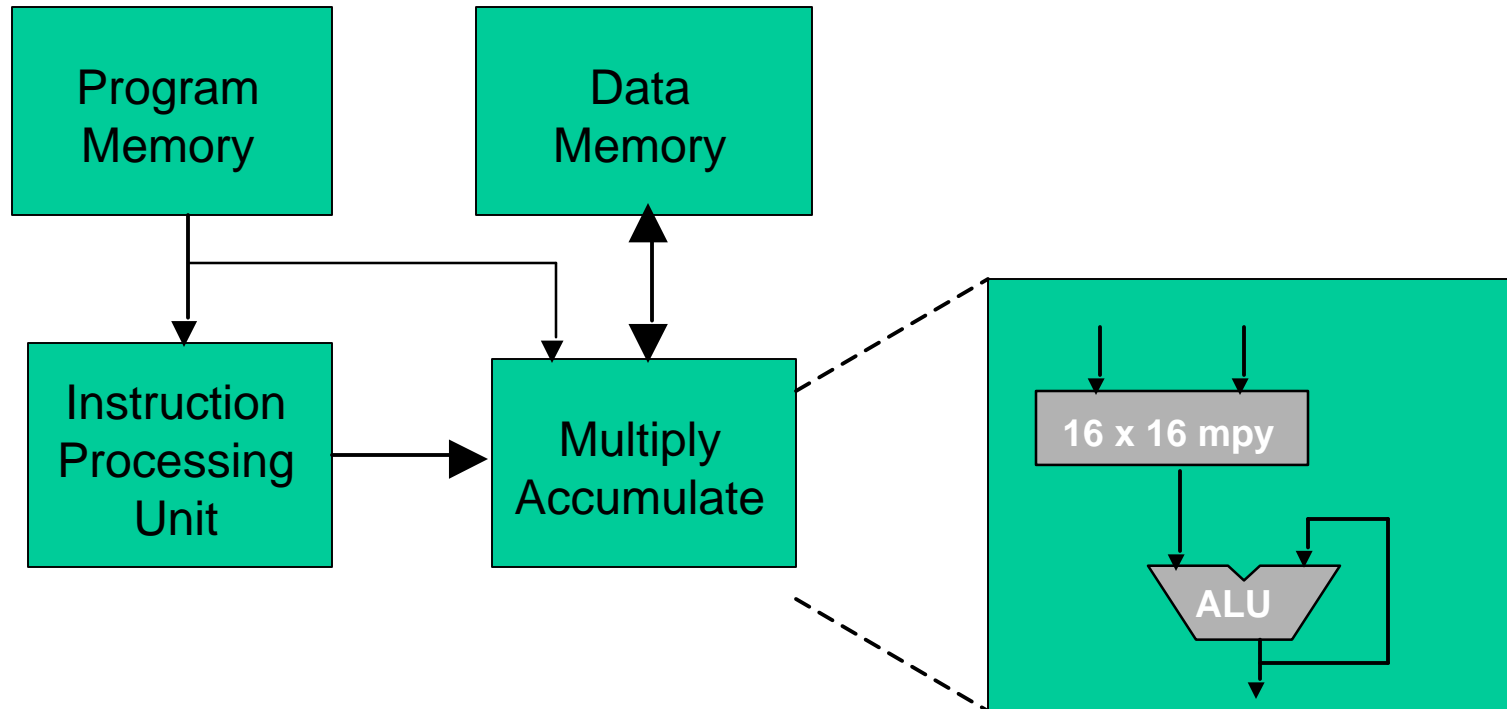
# TMS320C1x Memory and Buses



- **Single cycle reads and writes**
- Modified Harvard Architecture
  - Separate Program and Data Buses
  - "Bridge" between Program and Data Space
- Up to 8K words of on-chip Program ROM  
4K words of
- EPROM and OTP available
- Up to 64K words External Program Memory

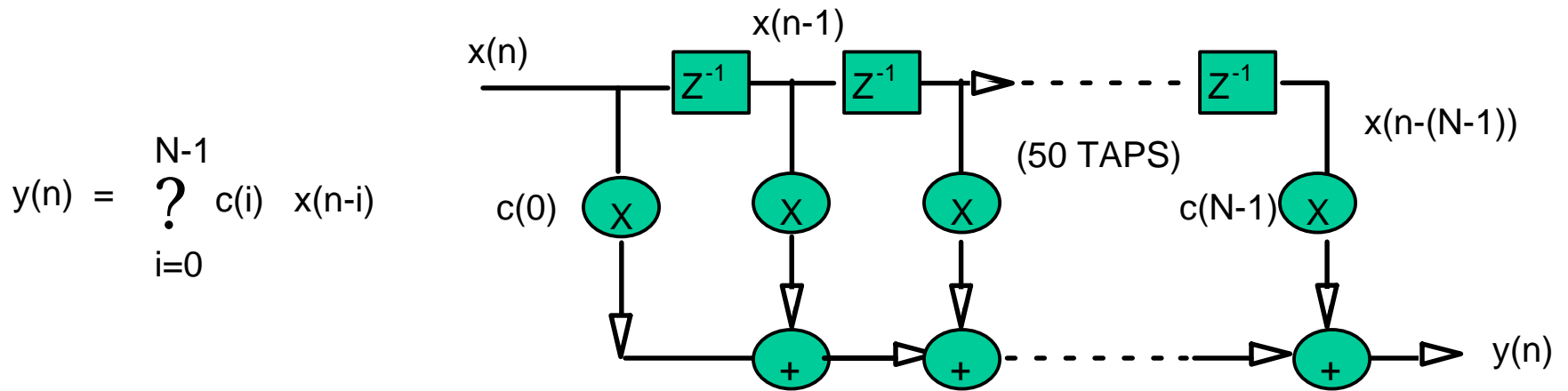
Courtesy: Texas Instruments

# Modified Harvard Architecture

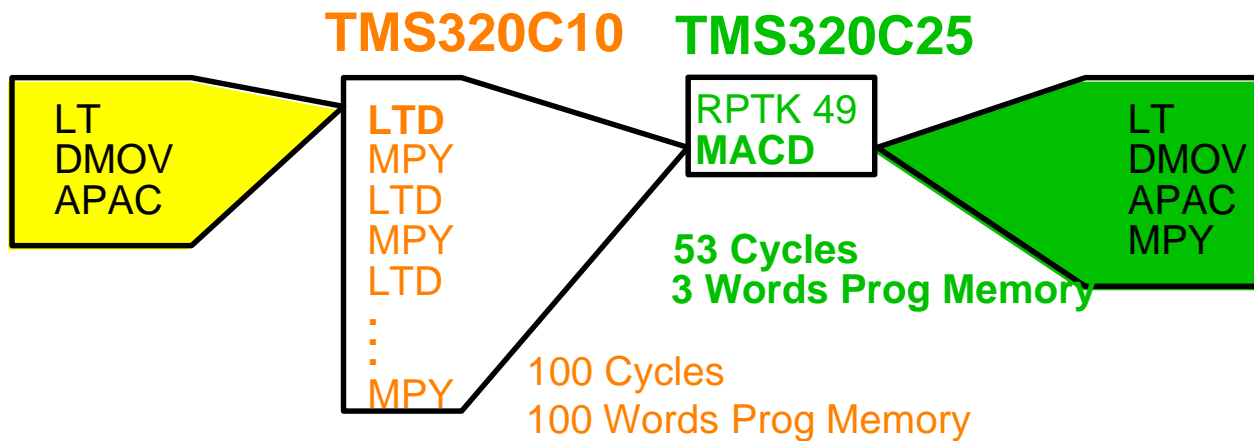


Program bus to get instruction  
Or to get coefficients (often stored in ROM)

# Same FIR: 53 cycles, 3 prog words



*Single Cycle Multiply - Accumulate!*



# Example: MACD

MACD = Multiply by Program Memory and Accumulate with Delay  
(Instruction is still present in C54x and C55x)

MACD Smem, pmad, src  
Smem = data memory  
pmad = program address  
src = accumulator (A or B)

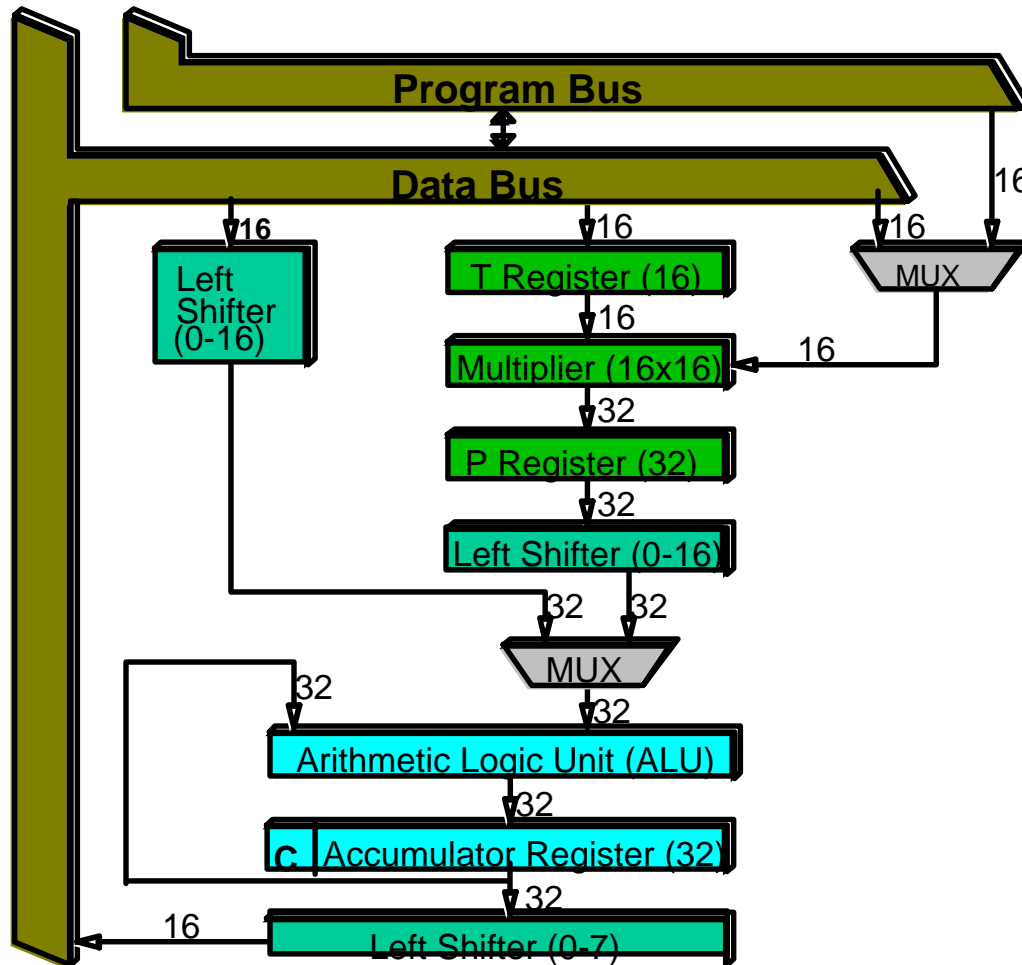
Executes (simplified):

(Smem) x (Pmem(at location pmad)) + src -> src	; = multiply – accumulate
(Smem) -> Treg	; load data in Treg register
(Smem) -> Smem +1	; load data in next mem loc.
(pmad) +1 -> pmad	; increment program address pointer

When executing with a repeat instruction, takes one cycle

# Single Cycle MAC

## TMS320C2x Multiplier/ALU



- Single Cycle 16x16 bit Multiply yielding a 32-bit product
- Supports simultaneous Program and two Data Operand acquisition
- Supports simultaneous ALU and Multiplier operations
- 0-16 bit Left Post-Shifter

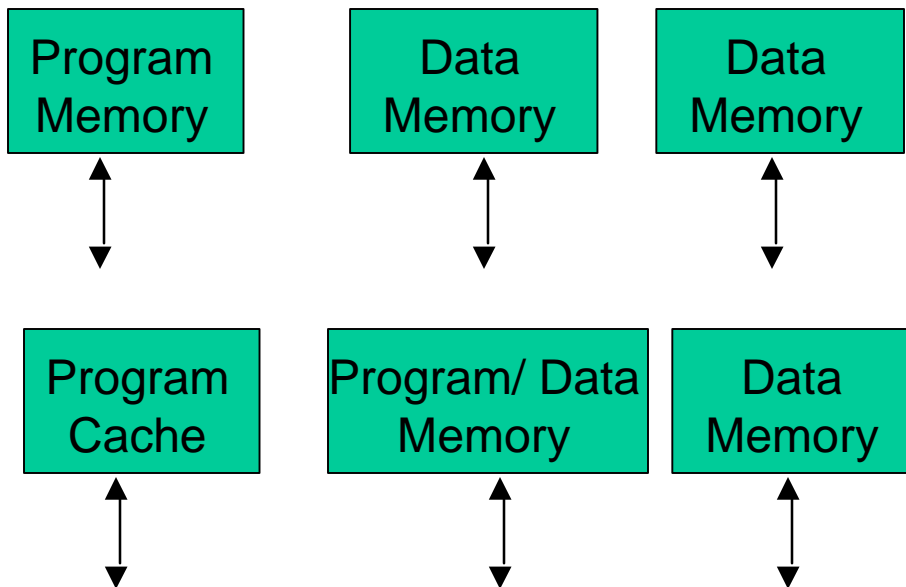
# TMS320C2x Enhancements Over C1x

1986:

- 80/100ns instruction cycle time
- Simultaneous single-cycle Multiply/ALU operations
- Zero overhead repeat single instruction
- 64K words of off-chip Data RAM
- Optimizing ANSI C-Compiler
- 544 words of on-chip Data/Program RAM
- Multiplier Post Shifter and enhanced Accumulator Post Shifter
- 74 additional instructions
  - Single-cycle MAC and zero overhead repeat
  - Long immediate and carry bit support
  - More logical and conditional branch operations
  - Data block move support
- Bit reversed addressing for FFTs
- Eight auxiliary registers
- Hardware wait states
- DMA support
- Idle and Powerdown Capability

Courtesy: Texas Instruments

## Other memory configurations



Multiple data memories  
e.g. Motorola 56000:  
- program memory  
- X memory  
- Y memory

### Instruction cache

- single instruction RPTK (repeat in TMS320C2x))
- a few instructions (up to 15 in AT&T 16A)
- **ALWAYS under programmers control!**
- **ALWAYS known at compile time!**

# Memory configurations (more)

- Very cost sensitive applications
  - all memory ON chip (even in the 80's!)
  - multiple small memories instead of unpredictable memory cache hierarchy
  - program memory mostly ROM (now Flash Memory)
  - Programmer decides the distribution of arrays over the memories to make sure that the two parallel reads are from different memory banks!
- 
- More fancy stuff:
    - special instructions to move samples in a delay line
    - circular buffers for delay lines

# Addressing modes

- $2^{16}$  memory locations
- only 16 bit instruction width means only one immediate address
- most processors: immediate address is two instruction words
  
- MOST used: register – indirect addressing
- very compact
- very useful for accessing consecutive memory locations in a repetitive mode
  
- Needs:
  - special address registers
  - associated Address calculation units
  - operate in parallel
  - as many ACU's as memories