Typical DSP architectures and features

extra materials

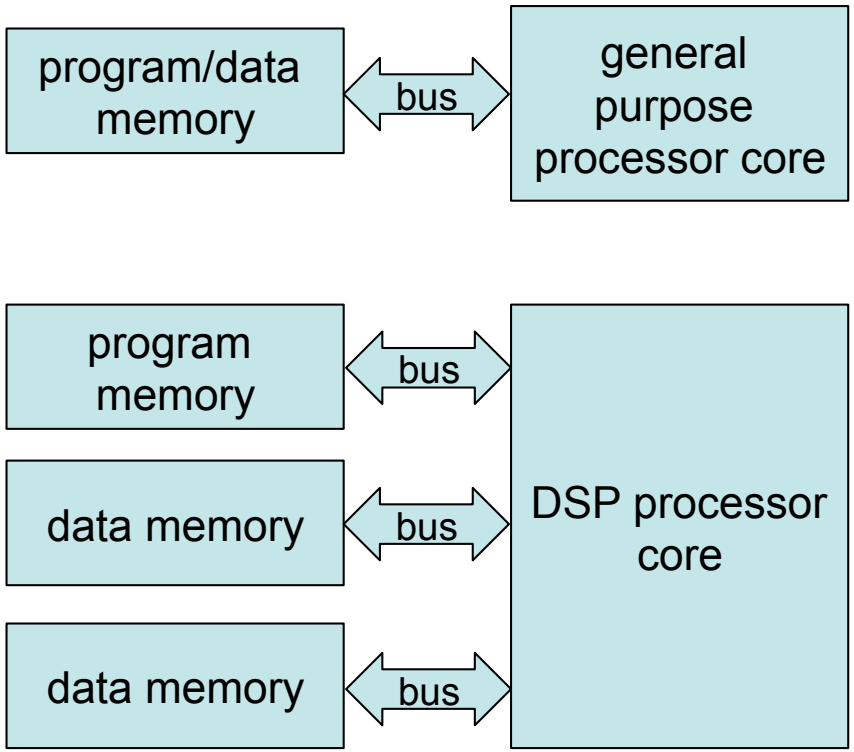
SS 2010
HW/SW Codesign

Christian Plessl

Classic DSP characteristics

- explicit parallelism
  - Harvard architecture for concurrent data access
  - concurrent operations on data and addresses
- optimized control flow and background processing
  - zero-overhead loops
  - DMA controllers
- special addressing modes
  - distinction of address, data and modifier registers
  - versatile address computation for indirect addressing
- specialized instructions
  - single-cycle hardware multiplier
  - multiply accumulate instruction (MAC)

Harvard architecture



- unified external memory for program and data
- all operands in registers

- separate program and data memories
- operands also in memory
- concurrent access to
  - instruction word
  - one or several data words
- example:

```
MPYF3   *(AR0)++,   *(AR1)++,   R0
```

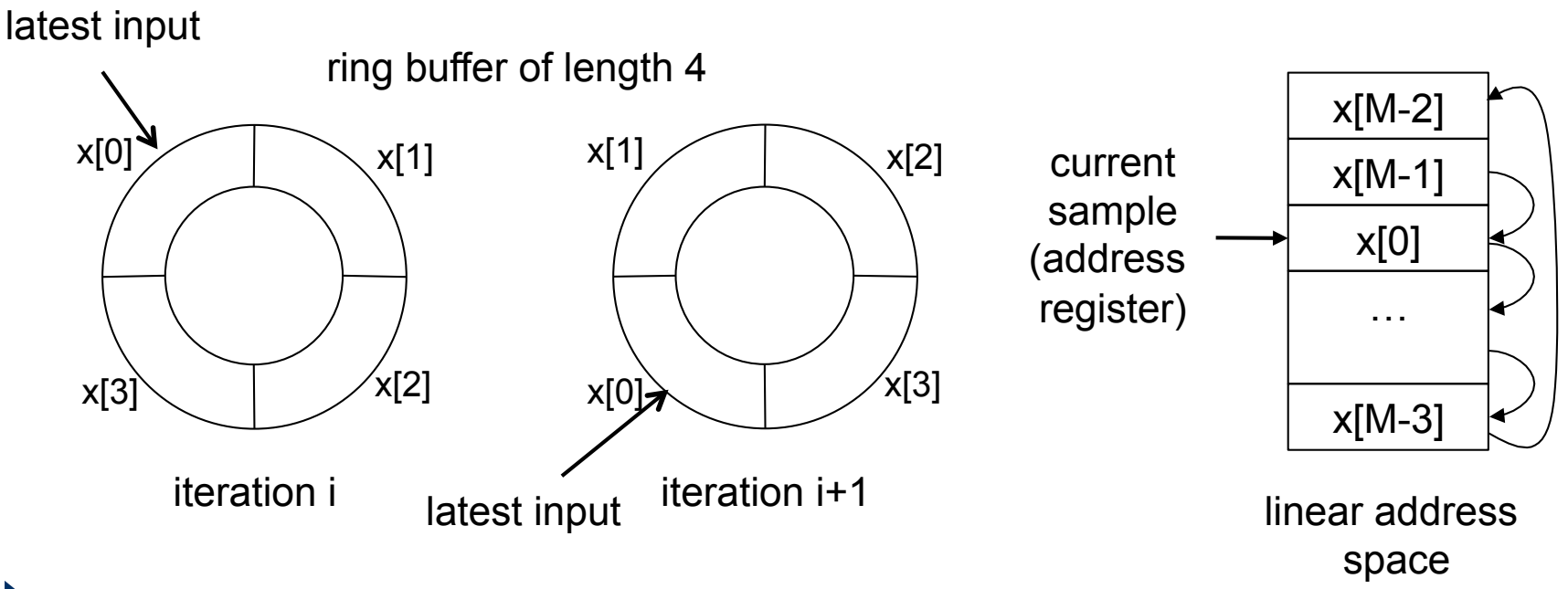
instruction from program memory	from data memory (address in address register AR0)	from data memory (address in address register AR1)	store result in data register R0

Specialized addressing modes

- many DSPs distinguish address registers from data registers
- additional ALUs for address computations
  - useful for indirect addressing (register points to operand in memory)
    `ADDF3  *(AR0)++, R1, R1`
  - operations on address registers in parallel with operations on data registers, no extra cycles
  - behavior depends on instruction and contents of special purpose registers (modifier registers)
- typical address update functions
  - increment/decrement by 1 (AR0++, AR0--)
  - increment/decrement by constant specified in modifier register (AR0 += MR0, AR0 -= MR5)
  - circular addressing (AR0 += 1 if AR0 < upper limit, else AR0 = base address), see example
  - bit-reverse addressing, see example
  - …

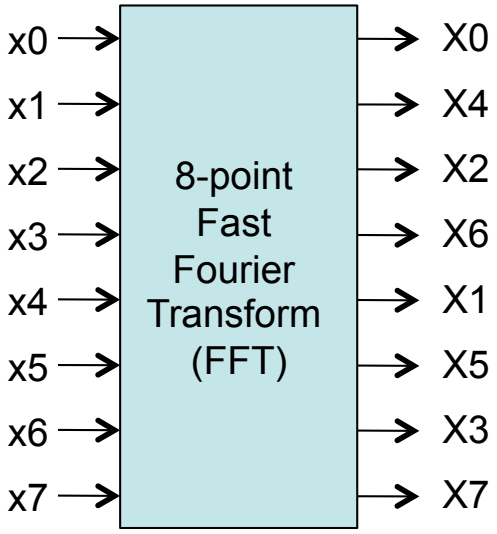
Circular addressing

- goal: implementation of ring buffers in linear address space
  - implementation variants
    - copy data with data access, or
    - use circular addressing (don't copy data, wrap pointers)
  - supported by addressing modes
    - data access and move operations
    - increment operators that wrap around at buffer boundaries



Bit-reverse addressing

- goal: accelerate FFT operation
- very important DSP operation
- transforms signals between time and frequency representations
- compute intensive:
  - N-point DFT needs O(N^2) complex multiplications
  - FFT needs O(N*log2(N)) complex multiplications



basic operation in many DSP algorithms

0 0 0	0 0 0 = 0
0 0 1	1 0 0 = 7
0 1 0	0 1 0 = 2
0 1 1	1 1 0 = 6
1 0 0	0 0 1 = 1
1 0 1	1 0 1 = 5
1 1 0	0 1 1 = 3
1 1 1	1 1 1 = 7

mirror bits (bit reverse)

reverse carry

```
  0 0 0  = 0
+ 1 0 0
-------
  1 0 0  = 4
+ 1 0 0
-------
  0 1 0  = 2
+ 1 0 0
-------
  1 1 0  = 6
+ 1 0 0
-------
  0 0 1  = 1
```

other method to compute addresses, add N/2 with reverse carry arithmetic

Zero-overhead loops

- goal
  - reduce overhead for executing loops
  - general purpose processors
    - initialize loop counter
    - execute loop body
    - check loop exit condition
    - branch to loop start or exit loop
  - digital signal processors
    - initialize loop counter
    - execute loop body
    - ~~check loop exit condition~~
    - ~~branch to loop start or exit loop~~

example: add first 100 values in array a and store result in R1

TMS320C3x-like assembler

```
LDI     @a, AR0
LDI     0.0, R1
RPTS    99
ADDF3   *(AR0)++, R1, R1
…
```

RPTS N repeats next instruction N-1 times

Putting it together: scalar product

```
sum = 0.0;
for (i=0; i<N; i++)
  sum = sum + a[i]*b[i];
```

TMS320C3x assembler

```
    LDI     @a, AR0
    LDI     @b, AR1
    LDF     0, R0
    LDF     0, R1
    RPTS    N-1
    MPYF3   *(AR0)++, *(AR1)++, R0
||  ADDF3   R0, R1, R1
    ADDF3   R0, R1, R1
```

data register

address register

zero-overhead loop

exploit harvard architecture, read two data operands in one cycle

MAC - instruction

address arithmetic (auto increment)

Further reading

- Jennifer Eyre and Jeff Bier, “The Evolution of DSP Processors”, BDTI Whitepaper
- Phil Lapsley et al., “DSP Processor Fundamentals”, IEEE Press
- Berkeley Design Technologies Website, http://www.bdti.com/