

EE201A Presentation

Memory Addressing Organization for Stream-Based Reconfigurable Computing

Team member:

Chun-Ching Tsan : Smart Address Generator
- a Review

Yung-Szu Tu : TI DSP Architecture and
Data address

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

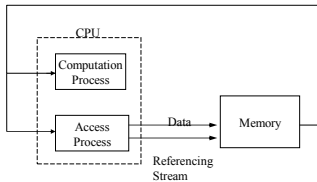
Outline – Smart Address Generator

1. Structured Memory Access (SMA) Machine (1983)
2. Application-specific Address Generator (ASAG) (1989)
3. Address Generation Unit (AGU) (1991)
4. GAG (generic address generator) (1990)
5. GAG of MoM-2 (1991)
6. GAG of MoM-3 (1993~1999)

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

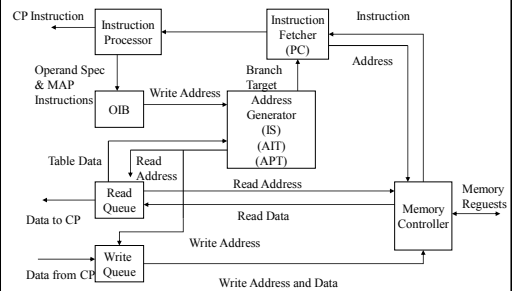
Structured Memory Access (SMA) Machine (1983)

CPU-Memory Model: a von Neumann machine
- computational processor (CP)
- memory access processor (MAP)



EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

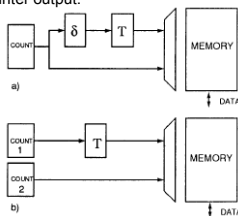
MAP Internal Organization



EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

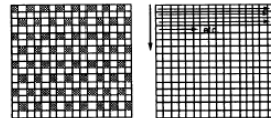
Application-specific Address Generator(ASAG) (1989)

- The needed address patterns are generated by a dedicated counters or circuit transformations applied to a counter output.



EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

A Simple Example for Image Processing

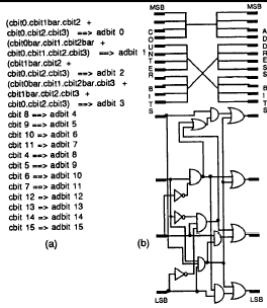


```

for Y = 0 to 65535 step 4096, (block height = 16 rows)
  for X = 0 to 255 step 16, (block width = 16 columns)
    for i = 1 to 4, (do 4 times)
      for y = 0 to 4095 step 512, (every 2nd line)
        for x = (y / 512) mod 2 to 15 step 2, (every 2nd pixel)
          address = x + y + X + Y,
          next x,
          next y,
          next i,
          next X,
          next Y
  
```

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Logic Synthesis for Semi-Random Address Sequences



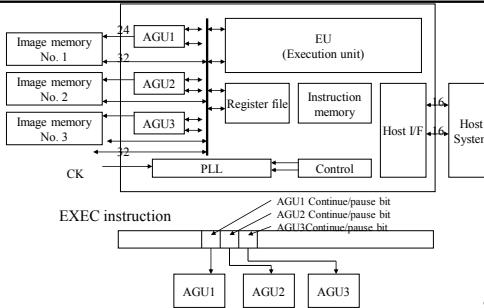
EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Address Generation Unit (AGU) (1991)

- an application specific address generation unit for video signal processor (VSP), a specified DSP
- implementing a 2-level address generation with window based memory access, without full slider method.
- 3 AGUs running in parallel calculate the address for external image memory
- Providing 17 addressing modes:
 - a 2-D raster scan mode
 - a block scan mode for spatial filtering
 - 8 variants of a neighborhood search mode
 - a 2-D indirect access mode for external image memory
 - a FFT mode and an affine transformation mode

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

DSP Architecture



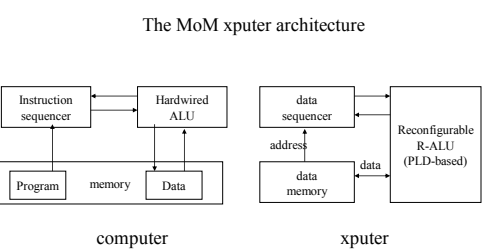
EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

GAG (generic address generator) (1990)

- MoM-1 (Map-oriented Machine 1)**
- an image processing machine with 2-D memory organization
- implement a pattern matching approach
- avoiding address calculation overhead and fully parallelized pattern matching by a dynamically reconfigurable PLA (DPLA)
- address generator: move control unit (MCU)
 - an application specific generic address generator
 - configured before execution time
 - needs no memory cycles at run time

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

GAG (generic address generator) (1990)



EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

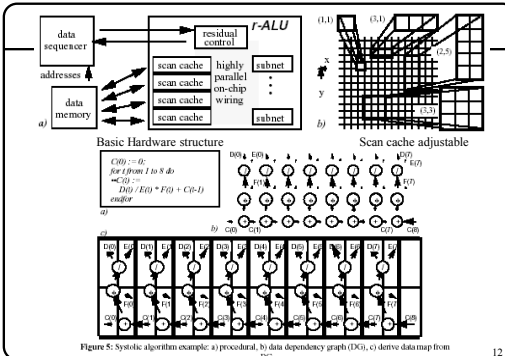
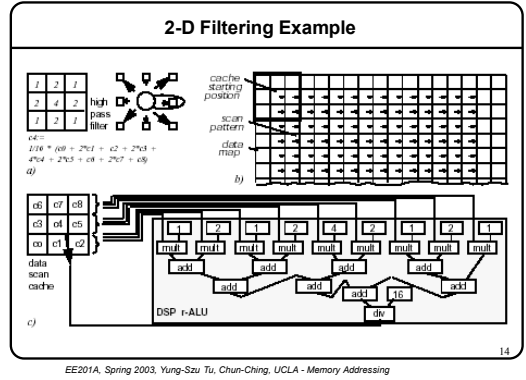
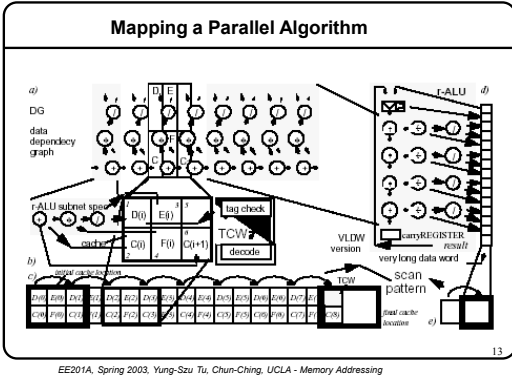


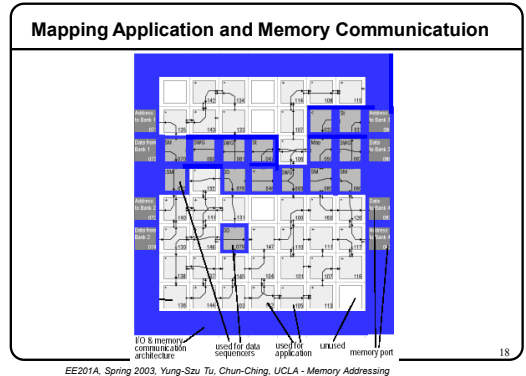
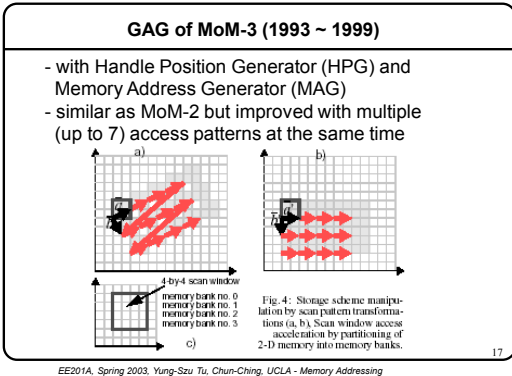
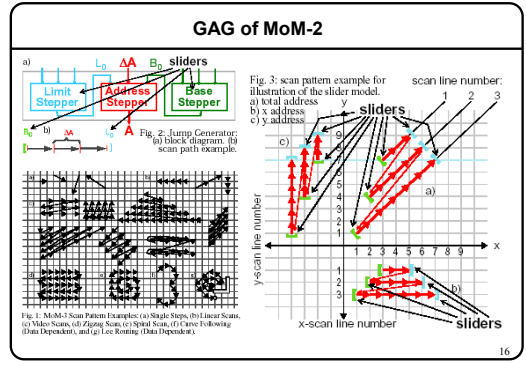
Figure 5: Synoptic algorithm example: a) procedural, b) data dependency graph (DAG), c) derive data map from DAG



GAG of MoM-2 (1991)

- 2 level address generator based on a flexible slider method
- configured by parameters and needs no memory cycles at run time
- Consists:
 1. Jump Generator
 2. Task Manager
 3. Single Step Control Unit(SSCU) - pipeline

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing



Texas Instruments TMS320C54x DSP Architecture and Data Addressing

Class presentation of EE201A
May 16, 2003

19

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Agenda

- Architecture
- Block diagram
- Immediate addressing
- Absolute addressing
- Accumulator addressing
- Direct addressing
- Memory-mapped register addressing
- Stack addressing
- Indirect addressing
- Reference

20

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Architecture

- Advanced Harvard architecture
 - Separate data and program memory allows a high degree of parallelism
- CPU can read and write to a single block in the same cycle

21

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Block Diagram

- Memory Access
 - 4 internal bus pairs
 - C,D for data read
 - E for data write
 - P for program
- Others
 - 2 40-bit Accum.
 - 40-bit Barrel shifter
 - 40-bit ALU
 - 17x17b multiplier and 40b dedicated adder perform a non pipelined single-cycle MAC

22

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Immediate and Accumulator Addressing

- The instruction syntax contains the specific value of the operand
 - LD #80h, A
- Immediate values can be 3,5,8,9, or 16 bits in length

Figure 5-1, RPT Instruction With Short-Immediate Addressing

1 instruction word	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 1 1 1 0 1 1 0 0 0 8-bit constant
--------------------	---

Figure 5-2, RPT Instruction With 16-Bit-Immediate Addressing

2 instruction words	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 16-bit constant
---------------------	---

- Accumulator addressing Uses the accumulator as an address
 - READA Smem

23

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Absolute addressing

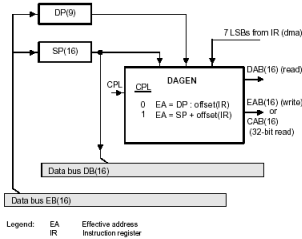
- Addresses are always 16 bits long, addressing types depend on instructions
- Data-memory address (dmdad) addressing uses a specific value to specify an address in data space
 - MVKD SAMPLE, *AR5
- Program-memory address (pmdad) addressing uses a specific value to specify an address in data space
 - MVPD TABLE, *AR7-
- Port address (PA) addressing uses a specific value to specify an external I/O port address
 - PORT FIFO, *AR5
- *(k) addressing uses a specific value to specify an address in data space
 - Instructions with single data-memory operand
 - LD *(BUFFER), A

24

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

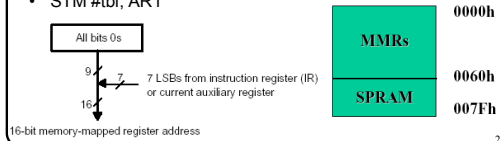
Direct addressing

- Uses the accumulator as an address
 - READA Smem
- With direct addressing, instructions contain the lower 7 bits of the data-memory address (dma)
 - Combined with a base address, data-page pointer (DP) or stack pointer (SP) to form a 16-bit data-memory address
 - ADD SAMPLE, B
 - DR-referenced
 - SP-referenced



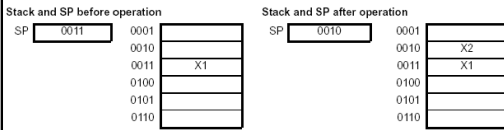
Memory-mapped register addressing

- Used to modify the memory-mapped registers without affecting the current data-page pointer (DP) or stack-pointer (SP)
 - Overhead for writing to a register is minimal
 - Works for direct and indirect addressing
 - SCRATCH-PAD ram LOCATED ON DATA PAGE 0 CAN BE MODIFIED
- STM #x, DIRECT
- STM #tbl, AR1



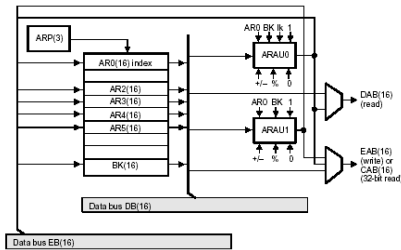
Stack addressing

- Used to automatically store the program counter during interrupts and subroutines
- Can be used to store additional items of context or to pass data values
- Uses a 16-bit memory-mapped register, the stack pointer (SP)
- PSHD X2



Indirect addressing

- 8 auxiliary registers (AR), and 2 auxiliary register arithmetic units (ARAU)

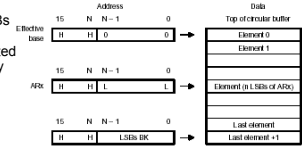


Indirect addressing (cont'd)

MOD Field	Operand Syntax	Function	Description ¹
0000 (0)	*ARx	addr = ARx	ARx contains the data-memory address.
0001 (1)	*ARx-	addr = ARx ARx = ARx - 1	After access, the address in ARx is decremented. ¹
0010 (2)	*ARx+	addr = ARx ARx = ARx + 1	After access, the address in ARx is incremented. ¹
0011 (3)	**ARx	addr = ARx + 1 ARx = ARx + 1	Before its use, the address in ARx is incremented; this new address is used to address the data-memory operand. ¹
0100 (4)	*ARx-0B	addr = ARx ARx = B(ARx - AR0)	After access, AR0 is subtracted from ARx with reverse carry (rc) propagation.
0101 (5)	*ARx-0	addr = ARx ARx = ARx - AR0	After access, AR0 is subtracted from ARx.
0110 (6)	*ARx+0	addr = ARx ARx = ARx + AR0	After access, AR0 is added to ARx.
0111 (7)	*ARx+0B	addr = ARx ARx = B(ARx + AR0)	After access, AR0 is added to ARx with reverse carry (rc) propagation.
1000 (8)	*ARx-%	addr = ARx ARx = circ(ARx - 1)	After access, the address in ARx is decremented using circular addressing. ¹
1001 (9)	*ARx-0%	addr = ARx ARx = circ(ARx - AR0)	After access, AR0 is subtracted from ARx using circular addressing.

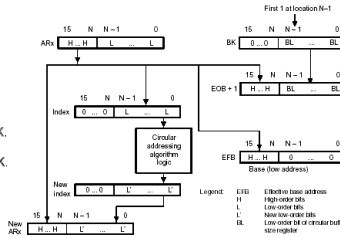
Indirect addressing (cont'd)

- Circular address modifications (MOD=8,9,10,11 or 14) for convolution, correlation, FIR filters, etc.
 - Circular buffer is a sliding window containing the most recent data
- Circular-buffer size register (BK) specifies the size of the circular buffer
 - Circular buffer of size R must start on a N-bit boundary, where $2^N > R$
 - 32-word circular buffer starts at xxxx xxxx xx00 0000
 - BK=32
 - Index is the N LSBs of ARx
 - Index is incremented or decremented by step



Indirect addressing (cont'd)

If $0 \leq \text{index} + \text{step} < \text{BK}$:
 $\text{index} = \text{index} + \text{step}$.
 Else if $\text{index} + \text{step} \geq \text{BK}$:
 $\text{index} = \text{index} + \text{step} - \text{BK}$.
 Else if $\text{index} + \text{step} < 0$:
 $\text{index} = \text{index} + \text{step} + \text{BK}$.



31

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

Indirect addressing (cont'd)

- Bit-Reversed Address Modifications (MOD=4 or 7)
 - Enhances execution speed and program memory for FFT algorithms that use a variety of radices
- Assume FFT size is 2^N , then $\text{AR0} = 2^{N-1}$
 - An ARx points to the physical location of a data value

Bit-Reversed Pattern	Bit-Reversed Step
*AR2+0B ; AR2 = 0110 0000 (0th value)	0000 0
*AR2+0B ; AR2 = 0110 1000 (1st value)	1000 8
*AR2+0B ; AR2 = 0110 0100 (2nd value)	0100 4
*AR2+0B ; AR2 = 0110 1100 (3rd value)	1100 12
*AR2+0B ; AR2 = 0110 0010 (4th value)	0010 2
*AR2+0B ; AR2 = 0110 1010 (5th value)	1010 10
*AR2+0B ; AR2 = 0110 0110 (6th value)	0110 6
*AR2+0B ; AR2 = 0110 1110 (7th value)	1110 14

32

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing

References

- Michael Herz, R. Hartenstein, M. Miranda: Memory Addressing Organization for Stream-Based Reconfigurable Computing; ICES 2002, pp. 815-817, 2002
- A. Pleszkun, E. Davidson: Structured Memory Access Architecture; Proceedings of IEEE International Conference on Parallel Processing, pp. 461-471, 1983.
- R. Hartenstein, A. Hirschbiel, M. Weber: A Novel Paradigm of Parallel Computation and its Use to Implement Simple High Performance Hardware; InfoJapan 90 - International Conference commemorating the 30th Anniversary of the Computer Society of Japan, Tokyo, Japan, 1990.
- D. Grant, P. Denyer, I. Finlay: Synthesis of Address Generators; Proceedings of IEEE International Conference on Computer-Aided Design (ICCAD), pp 116-119, 1989.
- K. Kitagaki, T. Oto, T. Demura, Y. Araki, T. Takada: A New Address Generation Unit Architecture for Video Signal Processing; Proceedings of SPIE International Conference on Visual Communications and Image Processing'91: Image Processing, Part Two of Two Parts, pp.891-900, Boston, MA, USA, Nov. 11-13, 1991
- Texas Instruments TMS320C54x DSP Reference Set, Volume 1: CPU and Peripherals (SPRU131)
- Texas Instruments TMS320C54x DSP Reference Set, Volume 2: Mnemonic Instruction Set (SPRU172B)

33

EE201A, Spring 2003, Yung-Szu Tu, Chun-Ching, UCLA - Memory Addressing