

Power Optimization by Datapath Width Adjustment

Yasuura, Hiroto

Faculty of Information Science and Electrical Engineering, Kyushu University | System LSI
Research Center

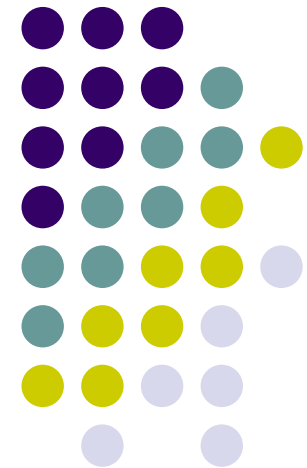
<https://hdl.handle.net/2324/9119>

出版情報 : SLRC プレゼンテーション, 2005-10-19. 九州大学システムLSI研究センター
バージョン :
権利関係 :



POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

Hiroto Yasuura
System LSI Research Center
Kyushu University





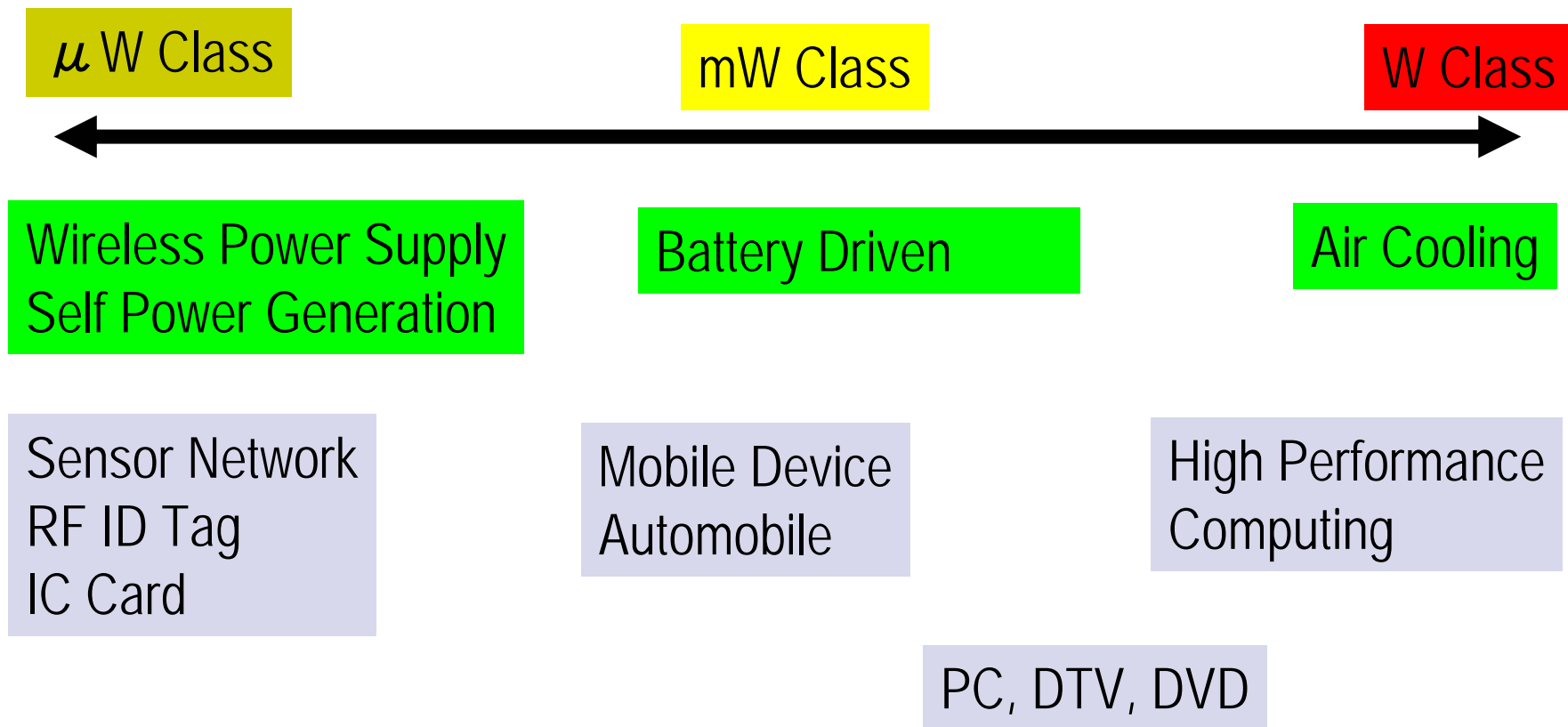
POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

- Introduction
- Datapath Width and Power Consumption
- Soft Core Processor
- Memory Width Optimization
- Shifted Computation
- Value-Based Clock Gating
- Conclusion

Low Energy SoC

- Increase of # of Transistors in a Chip
 - Dynamic Power Consumption (# of Switching)
 - Static Power Consumption (Leakage Power)
 - Memory, Analog, RF and MEMS
- Increase of Clock Frequency
- Applications in Mobile Devices
 - Battery Driven Systems
 - Battery Free Systems (RF ID Tags, Sensor Networks)
- Energy Limitation of the Earth

Ultra Low Power Design Technologies



Techniques for Energy Reduction

- Dynamic Energy Reduction
 - Voltage Scaling (Dynamic Voltage Scheduling)
 - Reduction of Capacitance
 - Reduction of # of Switchings
 - Clock Gating
 - Datapath Width Adjustment
- Static Energy (Leakage) Reduction
 - Threshold Control
 - New Device/Circuit Structure
 - Power Supply Gating
 - Datapath Width Adjustment



POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

- Introduction
- Datapath Width and Power Consumption
- Soft Core Processor
- Memory Width Optimization
- Shifted Computation
- Value-Based Clock Gating
- Conclusion



Datapath Width and Power Consumption

- The datapath width of the system
 - Buses, ALUs, Registers, Memories...
 - Determined by system designers or standard parts.
- Each application requires different accuracy of computation.
 - Specifications of input/output signals (Requirement of Quality)
 - Requests from algorithms (Requirement of Accuracy)
- Required datapath width is different from the width of HW.
- Larger datapath width for small bit-width data
 - Wasteful dynamic power in switching of the extra bits on the datapath.
 - Extra leakage power consumption



MPEG-2 Video Decoder

● Size of Program : 6275 lines (written in C)

Variable size
analysis result :

| Bit Width | # of variables | Bit Width | # of variables |
|-----------|----------------|-----------|----------------|
| 1 bits | 50 | 17 bits | 2 |

| Bit Width | Arrays |
|-----------|--------|
| 1 bits | 9 * 4 |

Variable Size Analysis Results
32 bits Variables * 100 = 35%

384 *int* type
variables

| | | | |
|---------|----|---------|----|
| 8 bits | 9 | 24 bits | 13 |
| 9 bits | 7 | 25 bits | 0 |
| 10 bits | 3 | 26 bits | 2 |
| 11 bits | 6 | 27 bits | 4 |
| 12 bits | 17 | 28 bits | 3 |
| 13 bits | 0 | 29 bits | 3 |
| 14 bits | 46 | 30 bits | 7 |
| 15 bits | 2 | 31 bits | 0 |
| 16 bits | 39 | 32 bits | 5 |

x * y : x is # of elements
y is # of arrays

Variable data-path Architecture

- Data-path which can be activated in 4-bit width.
- Variables A:16bits, B:4bits, C:12bits

| Ordinary Data-path | Variable Data-path |
|-----------------------|-----------------------|
| A 1100 1111 1100 0011 | A 1100 1111 1100 0011 |
| B 0000 0000 0000 0100 | B 1100 1111 1100 0100 |
| C 1111 1111 0010 1111 | C 1100 1111 0010 1111 |
| # of SW = 23 | # of SW = 9 |

Experimental Results

- Program
 - Split (File Split) and Sort in Unix
- Architecture
 - Variable Data-path Architecture Processor: VDP
 - 32bit RISC Processor: Fixed

| | # of Gates | Delay | Split | Sort |
|-----------------|------------|---------|---------|---------|
| VDP | 3,478 | 28.46ns | 60.57mW | 69.18mW |
| Fixed | 3,199 | 27.47ns | 72.39mW | 96.05mW |
| Power Reduction | | | -16.32% | -27.98% |



POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

- Introduction
- Datapath Width and Power Consumption
- **Soft Core Processor**
- Memory Width Optimization
- Shifted Computation
- Value-Based Clock Gating
- Conclusion

Reducing the lower bits of datapath



Changing Computation Quality

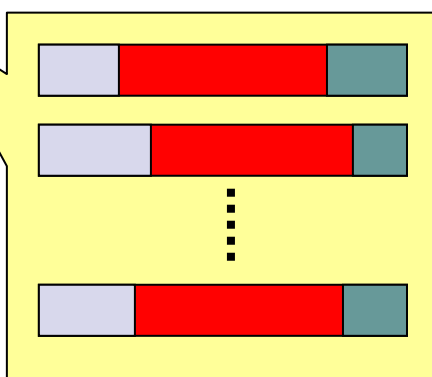
- When the accuracy of computation can be reduced, system designer can reduce the lower bits of variables.

Program

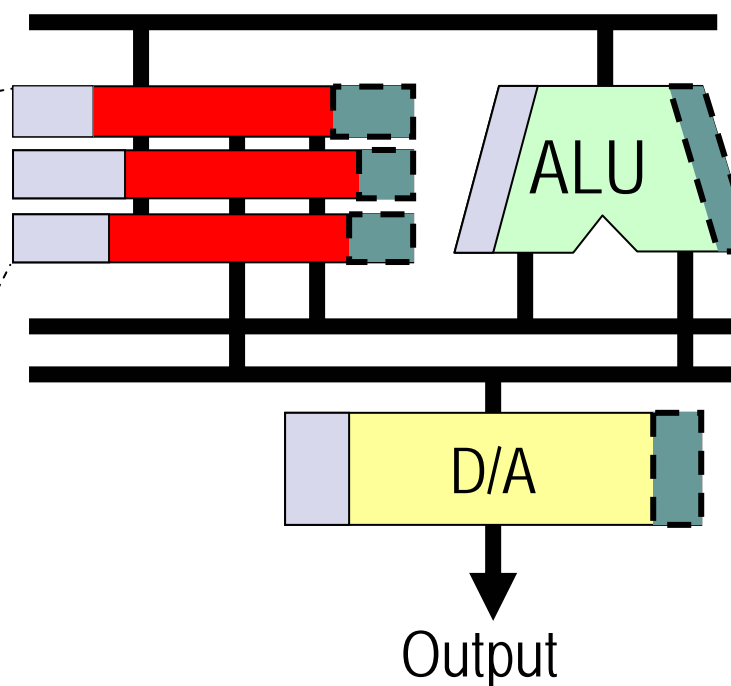
```
int func(v1, v2)
{
    int x0, x1, x2, x3, x3;
    char xdfgp, leergre;

    x0 = v1 + v2;
    x1 = v2 - v1;
    xdfgp = x0 * x1;
    if (x1 > x2) {
        leergre = x2 * x3;
        xdfgp = x3 - x1;
    } else {
        x1 = 1;
        x2 = xdfgp / x3;
    }
    while (x1 != 0) {
        leergre = x2 / 2;
        xdfgp = x3 / 5;
    }
}
```

Variables



Datapath



Reducing the upper bits of datapath



No Chang in Quality

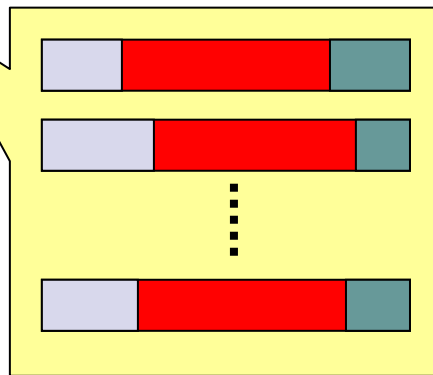
- It is difficult to know the range of the variables.

Program

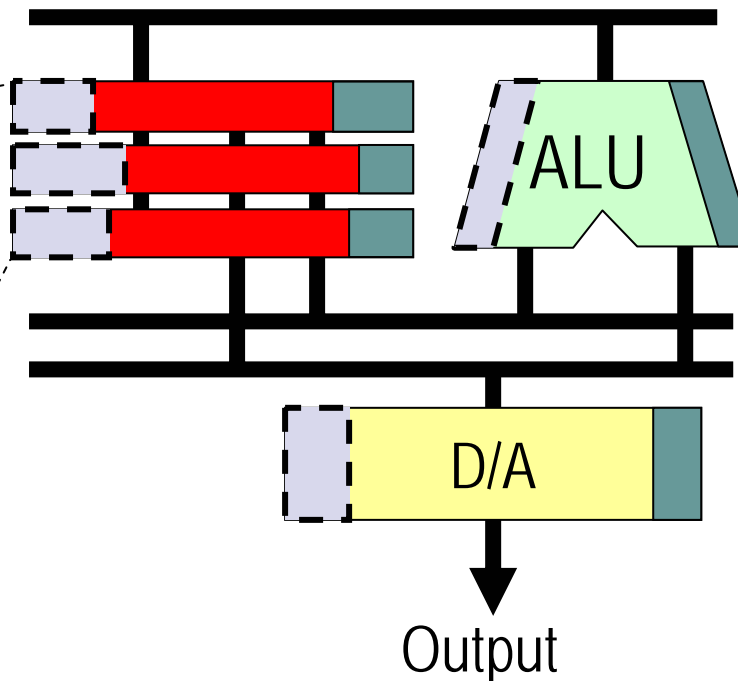
```
int func(v1, v2)
{
    int x0, x1, x2, x3, x3;
    char xdfgp, leergre;

    x0 = v1 + v2;
    x1 = v2 - v1;
    xdfgp = x0 * x1;
    if (x1 > x2) {
        leergre = x2 * x3;
        xdfgp = x3 - x1;
    } else {
        x1 = 1;
        x2 = xdfgp / x3;
    }
    while (x1 != 0) {
        leergre = x2 / 2;
        xdfgp = x3 / 5;
    }
}
```

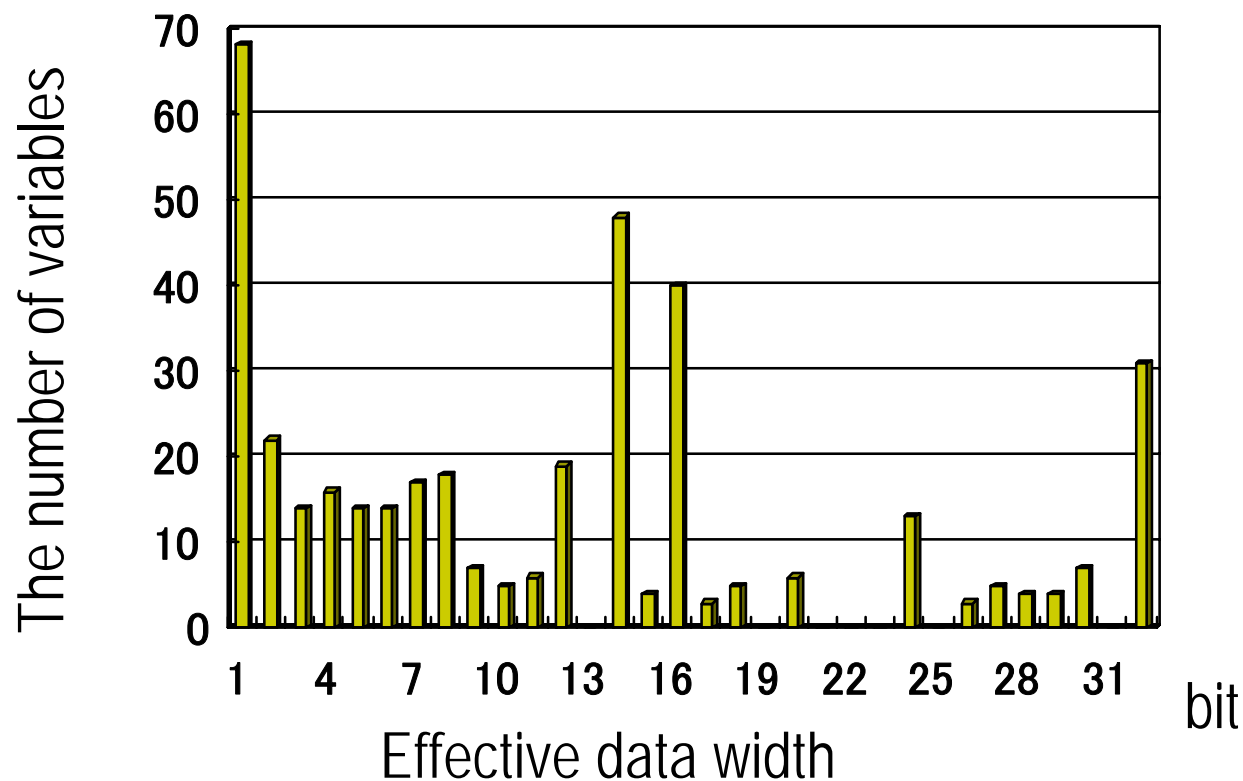
Variables



Datapath



Result of Variable Size Analysis

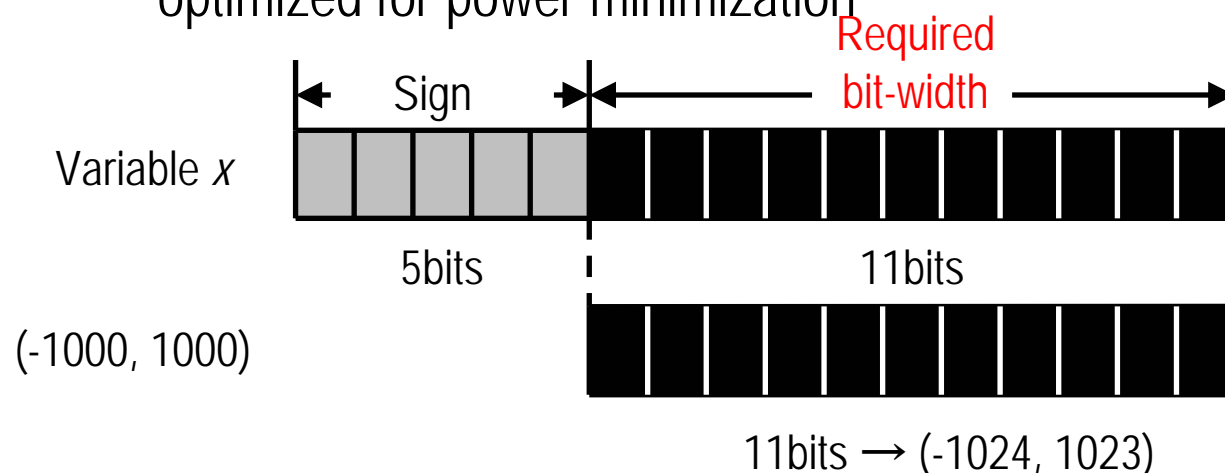


MPEG-2 video decoder

Variable Size Analysis

- *Concepts*

- Examine the range of variables and find the required bit-width for each variables.
- By analyzing required bit-width of variables, the datapath width can be optimized for power minimization



Kyushu Univ.
NAIST

MPEG2 Decoder:
05.10.19

Variable Size Analysis Results
32 bits Variables * 100 = 35%

Soft-core Processor

- Parameterized core processor
 - Design parameters
 - the datapath width
 - the number of general registers
 - Instruction set
 - Data/instruction memory space
- Logic and layout synthesis tools
- Retargetable compiler
- IP of customized core processor

Valen-C and a Retargetable Compiler

- Valen-C
 - Programmers can specify the effective bit width for each variable.
`int20 x, y, z`
 - The semantics of the program can be independent from processor architecture.
- Retargetable compiler
 - Processor Definition + Valen-C Program
⇒ Assembly code for the processor

Datapath Width and Data Memory

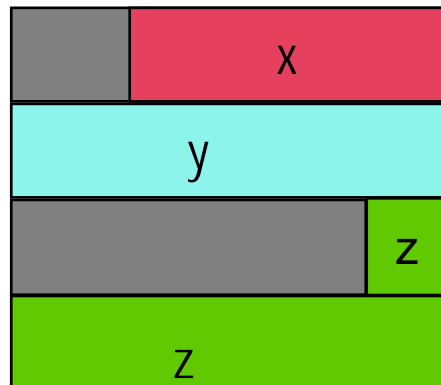
Valen-C Program

int12 x;

int20 y;

int24 z;

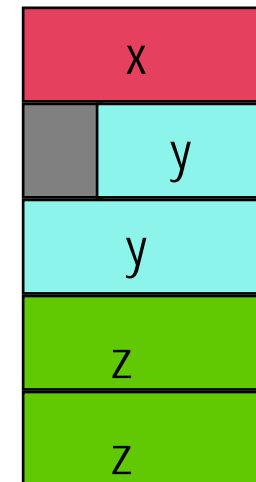
20-bit processor



unused: 24 bits

total: 80 bits

12-bit processor



unused: 4 bits

total: 60 bits

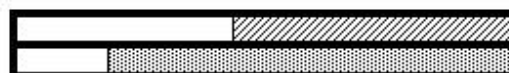


unused bits

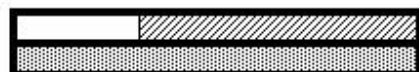
A Simple Example

```
main()
{
  int18 x;
  int26 y;
  .....
}
```

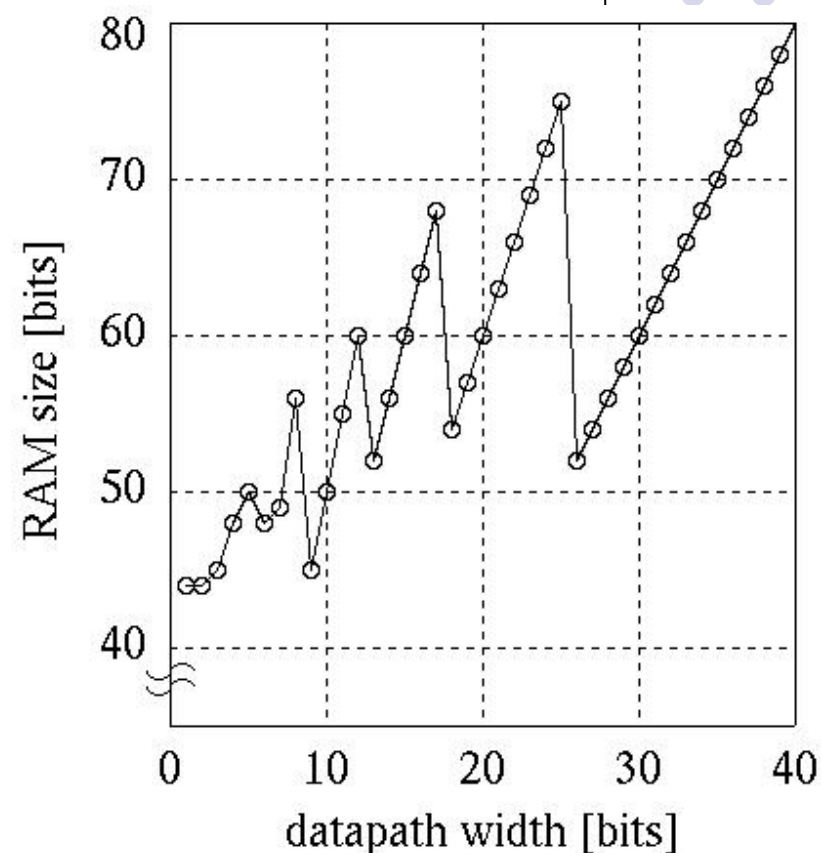
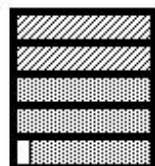
32 bits x 2 words = 64 bits



26 bits x 2 words = 52 bits



9 bits x 5 words = 45 bits



Trade-off between Area and Performance

```
int20 x, y, z;
z = x + y;
```

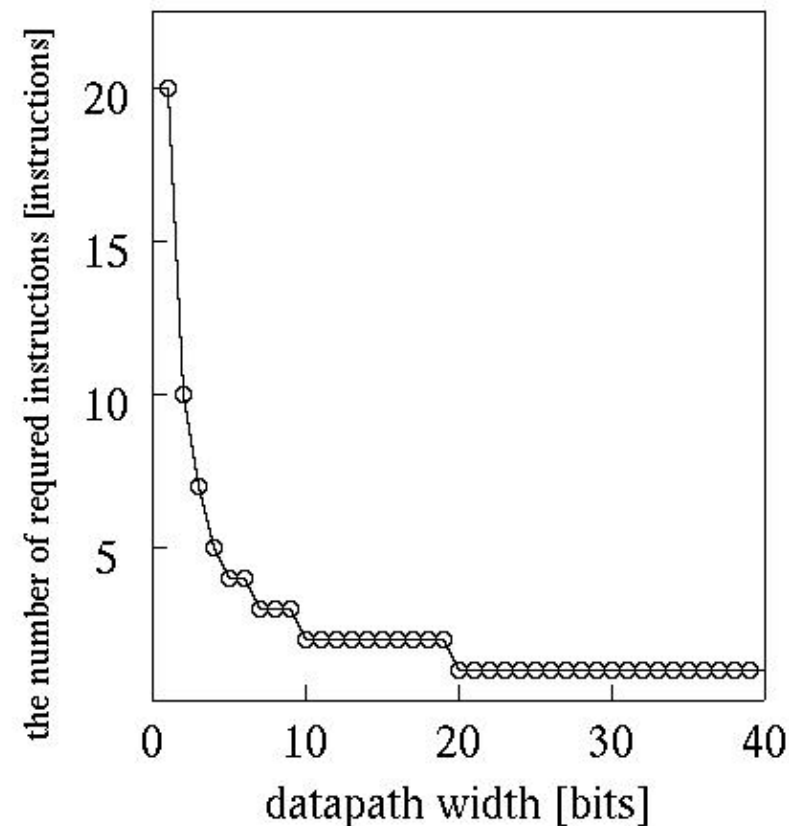
datapath width = 20 bits

```
add z, x, y
```

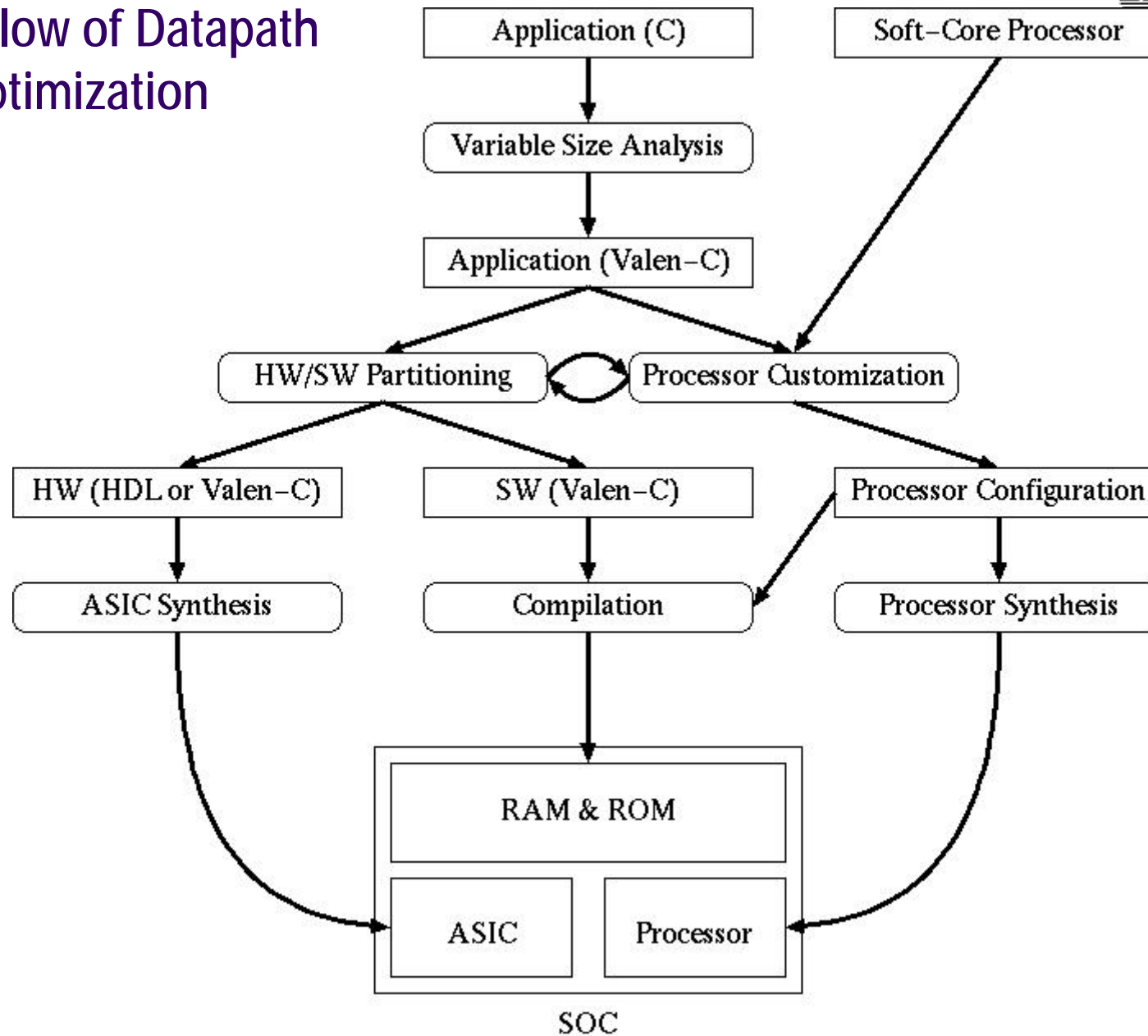
datapath width = 10 bits

```
add z_low, x_low, y_low
```

```
adde z_high, x_high, y_high
```



Design Flow of Datapath Width Optimization



Valen-C Compiler

Valen-C Program

```
main(){
    unsigned int1 flag;
    int14 x, y;
    int20 z, w;

    if (flag = 1) {
        z = x + y;
    }else{
        z = w;
    }
    ...
}
```

Machine Description File

Datapath width:10bits

short : 5 bits
int : 10 bits
long : 20 bits
long long : 30 bits

Valen-C Retargetable Compiler

Valen-C to C Translation

C Program

```
main(){
    unsigned short flag;
    long x, y;
    long z, w;

    if (flag = 1) {
        z = x + y;
    }else{
        z = w;
    }
    ...
}
```

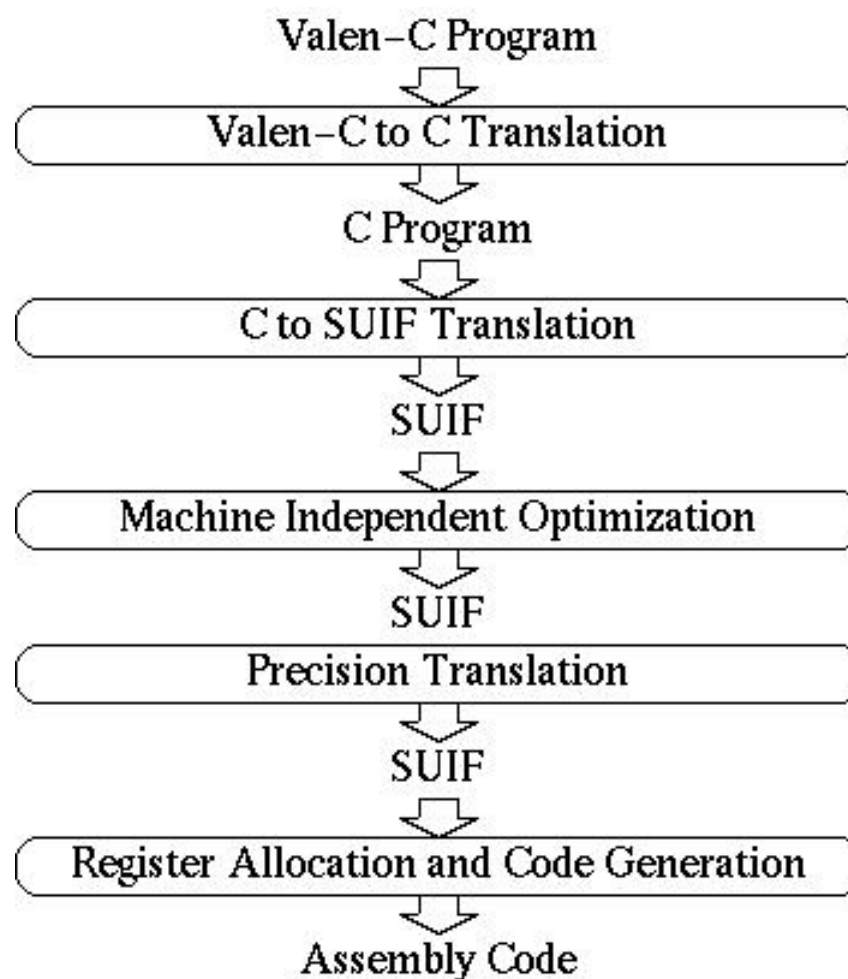
Code Generation

Assembly Code

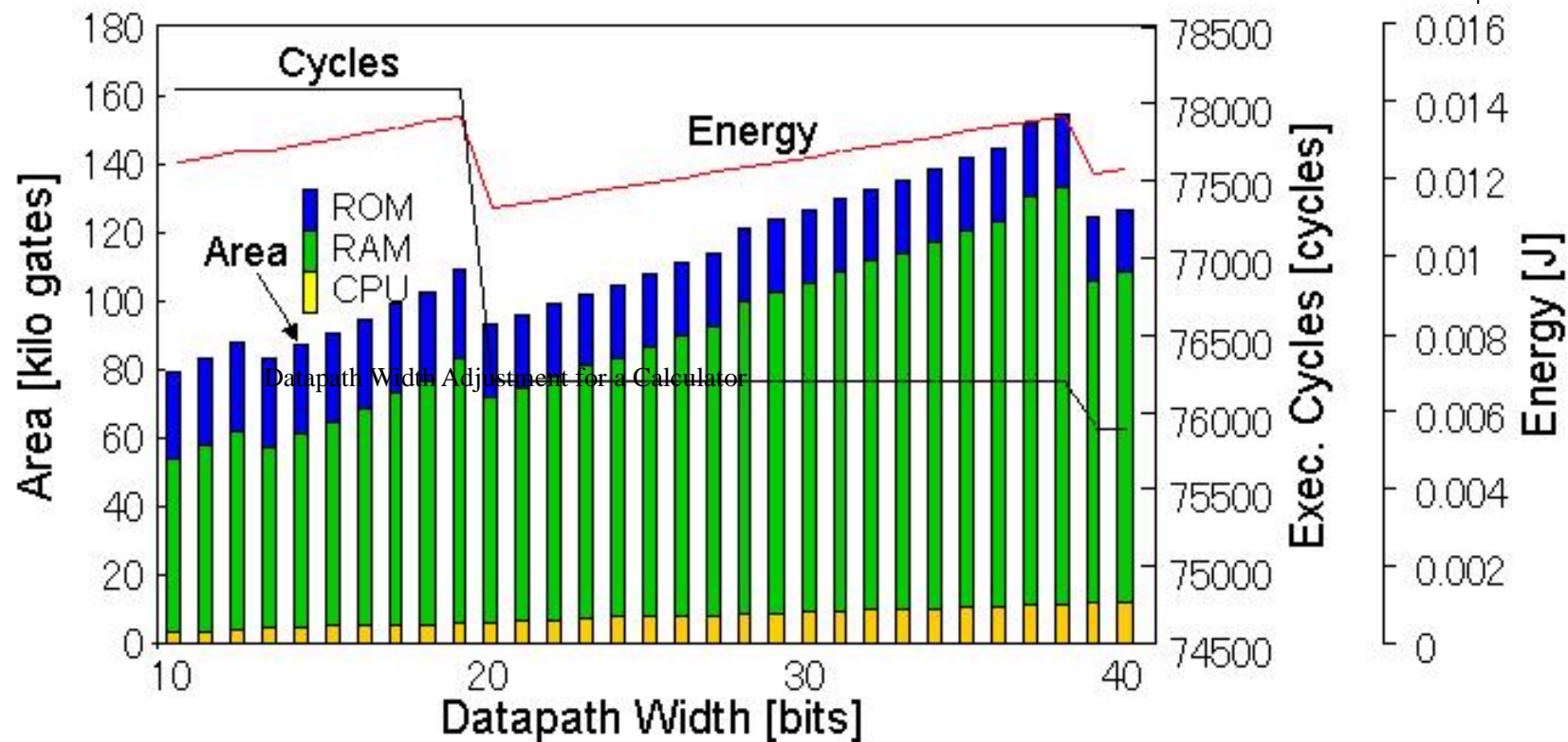
```
seq    tmp, flag, 1
beqz   tmp, L1
add    zl, xl, yl;
addc   zu, xu, yu;
jmp    L2:
L1: move    zl, wl;
      move    zu wu;
L2:
```

zl : lower bits of z
zu : upper bits of z
seq : set conditinal if two operands
are equal each other
addc : add with carry
beqz : branch equal zero

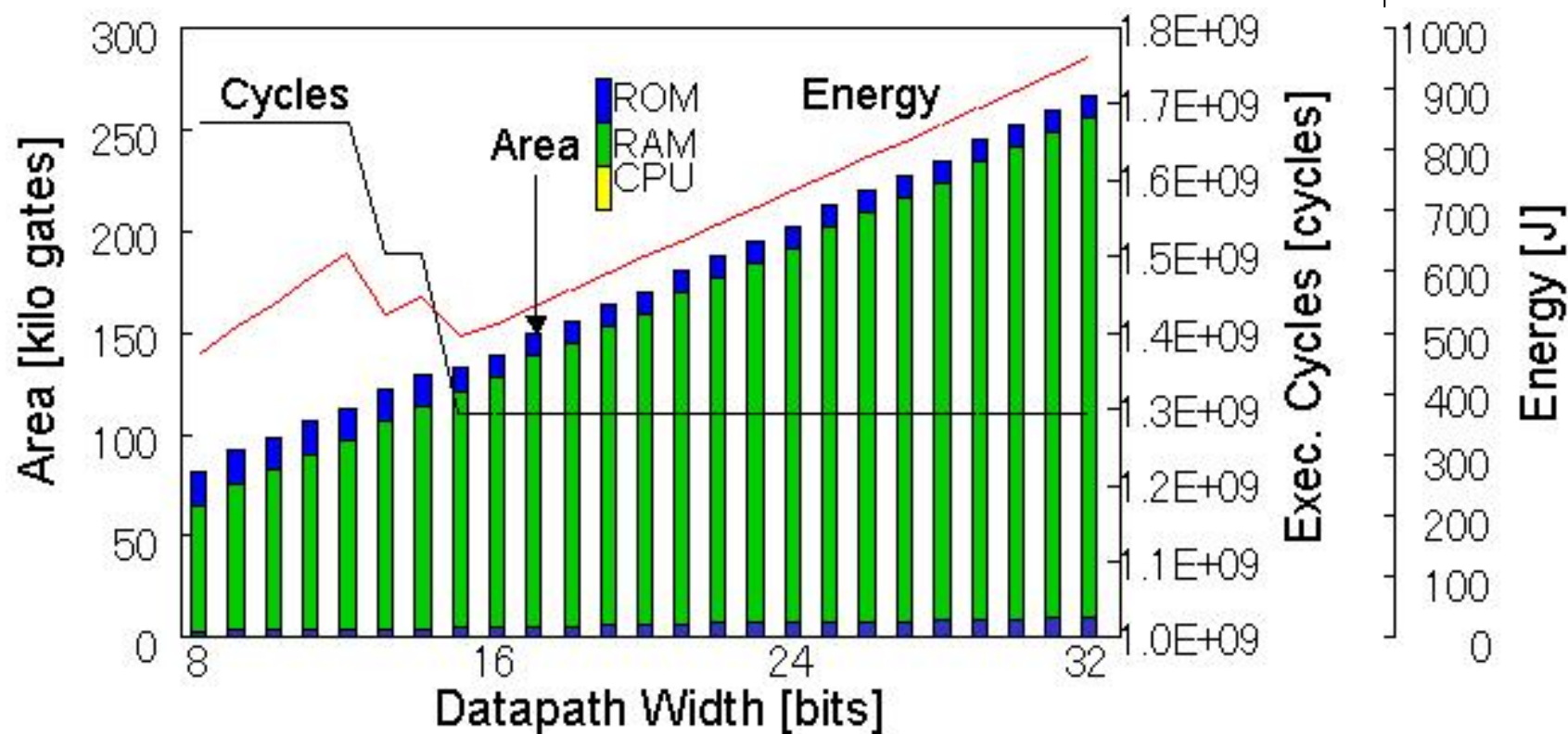
Flow of Valen-C Compiler



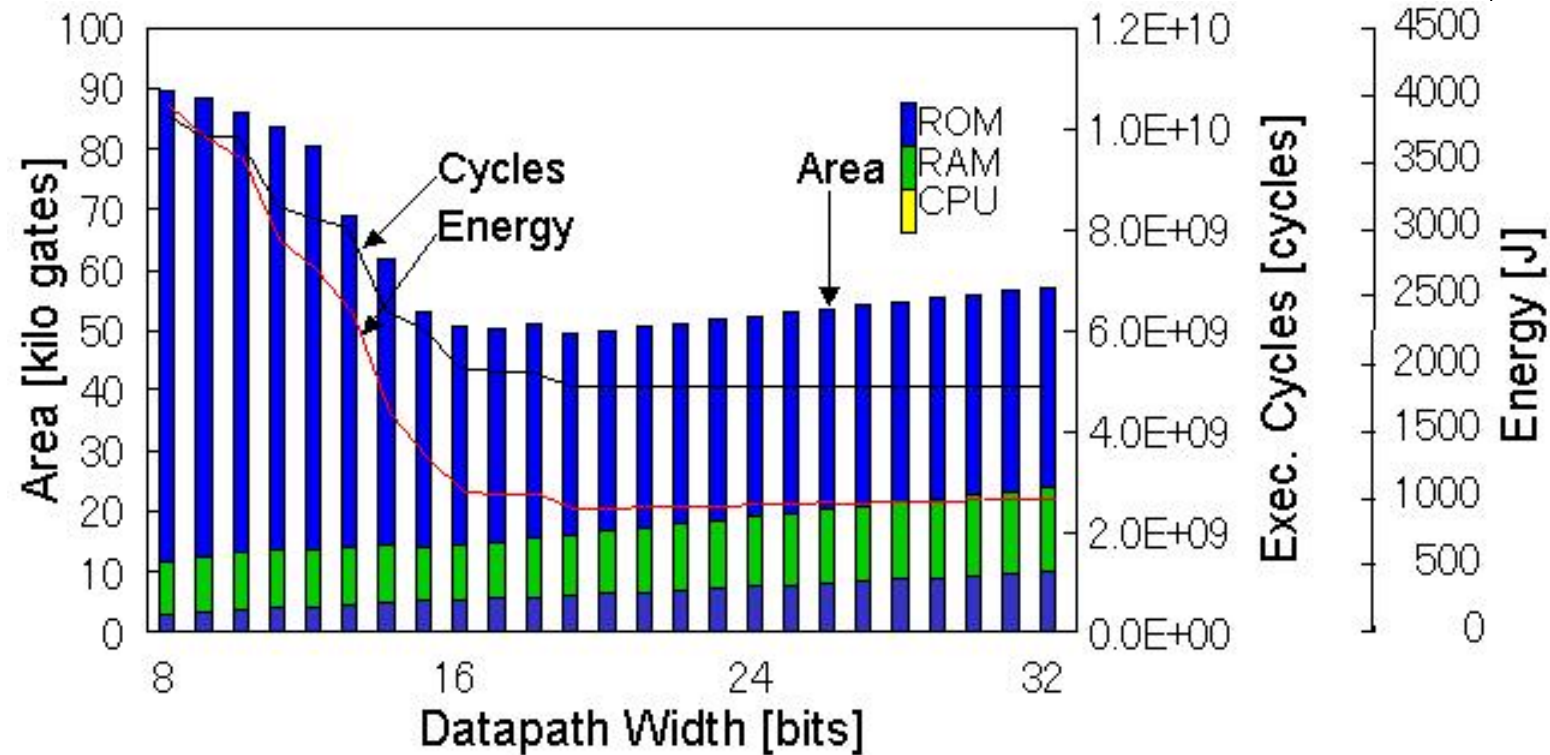
Datapath Width Adjustment for a Calculator



Datapath Width Adjustment for Lempel-Ziv encoder/decoder

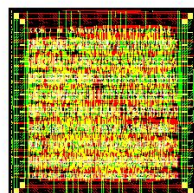
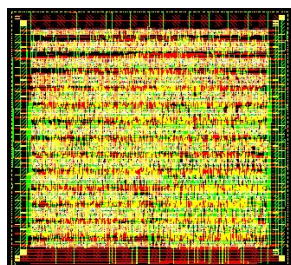


Datapath Width Adjustment for ADPCM encoder



Example of *Datapath Width Optimization*

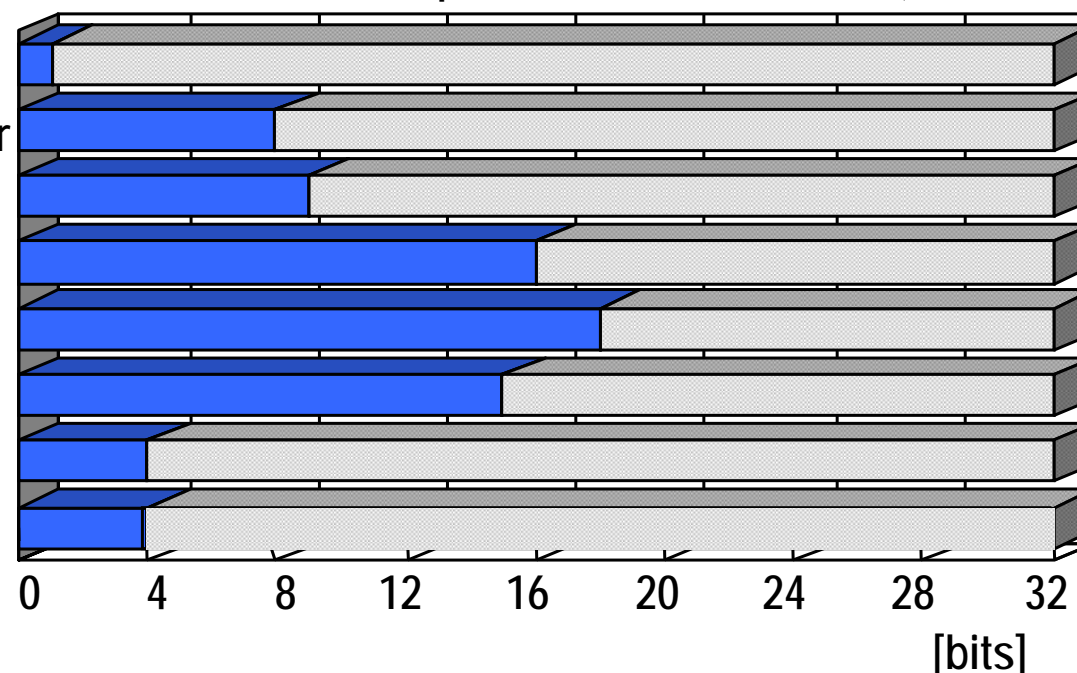
ADPCM decoder



[Variables]

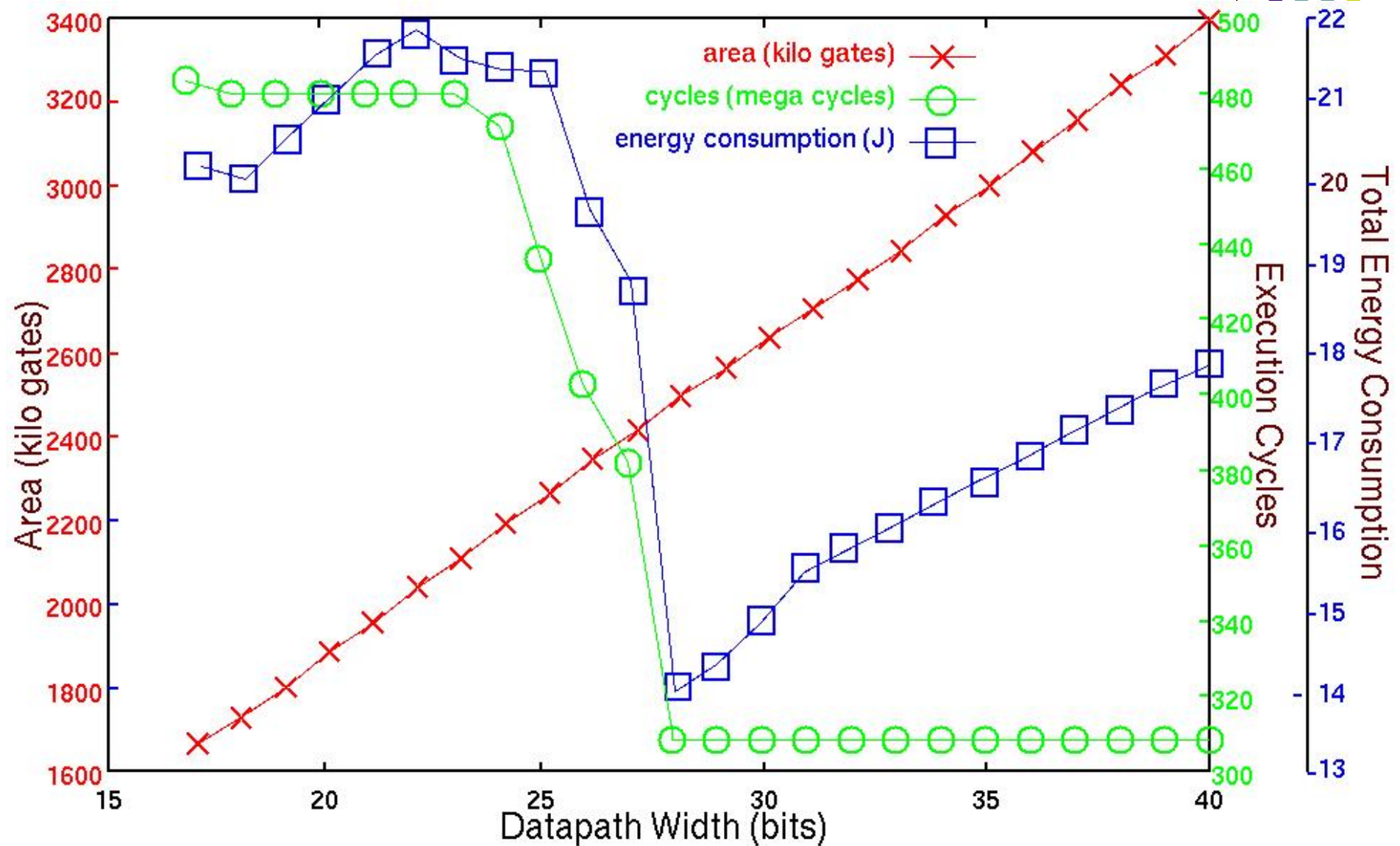
bufferstep
inputbuffer
index
vpdiff
valpred
step
delta
sign

Required bit width



| | ADPCM32 | ADPCM18 | Reduction |
|--------------|----------------------------------|--------------------------------|-----------|
| Area | 1220.8×1196.0 [um ²] | 865.2×865.2 [um ²] | 49% |
| #Cells | 1379 | 669 | 52% |
| #Transistors | 13006 | 5864 | 55% |
| Energy | 367 [nJ] | 239 [nJ] | 35% |

Datapath Width Adjustment for MPEG2 decoder



Point 1

- Datapath width adjustment can reduce power consumption without loss of performance and quality.
- Key techniques
 - Variable size analysis
 - Valen-C and Its compiler
 - HW/SW co-design tools
 - Softcore Processor
- Optimization of datapath width
 - At most 50% power reduction for practical applications

Datapath Width Adjustment



Quality-Driven Design

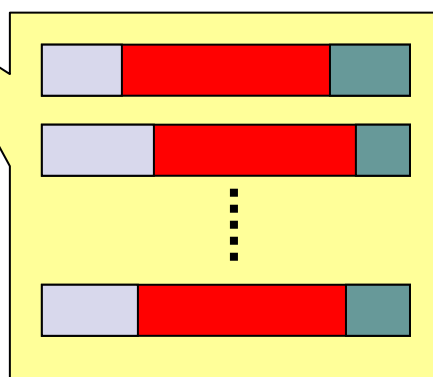
- Adjust datapath width for required quality.
- Reduce both upper and lower bits, if possible.

Program

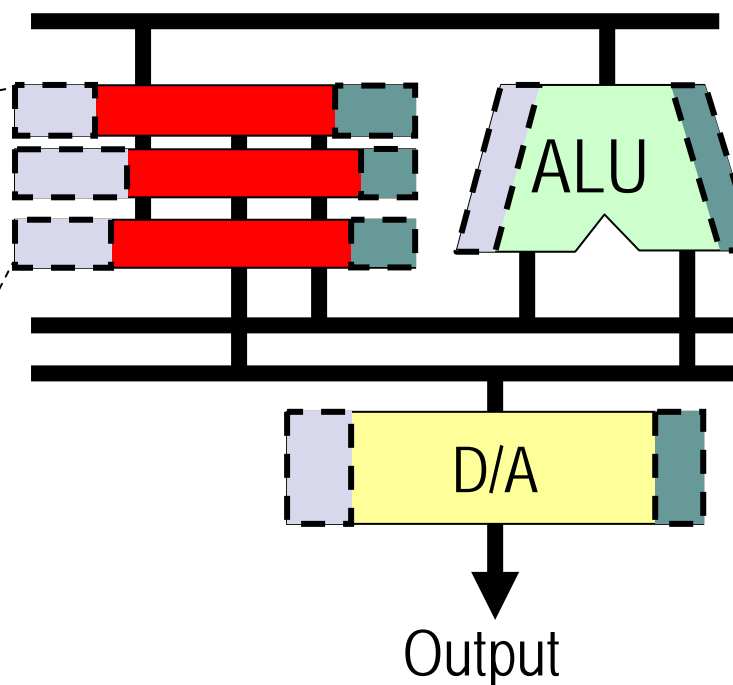
```
int func(v1, v2)
{
    int x0, x1, x2, x3, x3;
    char xdfgp, leergre;

    x0 = v1 + v2;
    x1 = v2 - v1;
    xdfgp = x0 * x1;
    if (x1 > x2) {
        leergre = x2 * x3;
        xdfgp = x3 - x1;
    } else {
        x1 = 1;
        x2 = xdfgp / x3;
    }
    while (x1 != 0) {
        leergre = x2 / 2;
        xdfgp = x3 / 5;
    }
}
```

Variables



Datapath



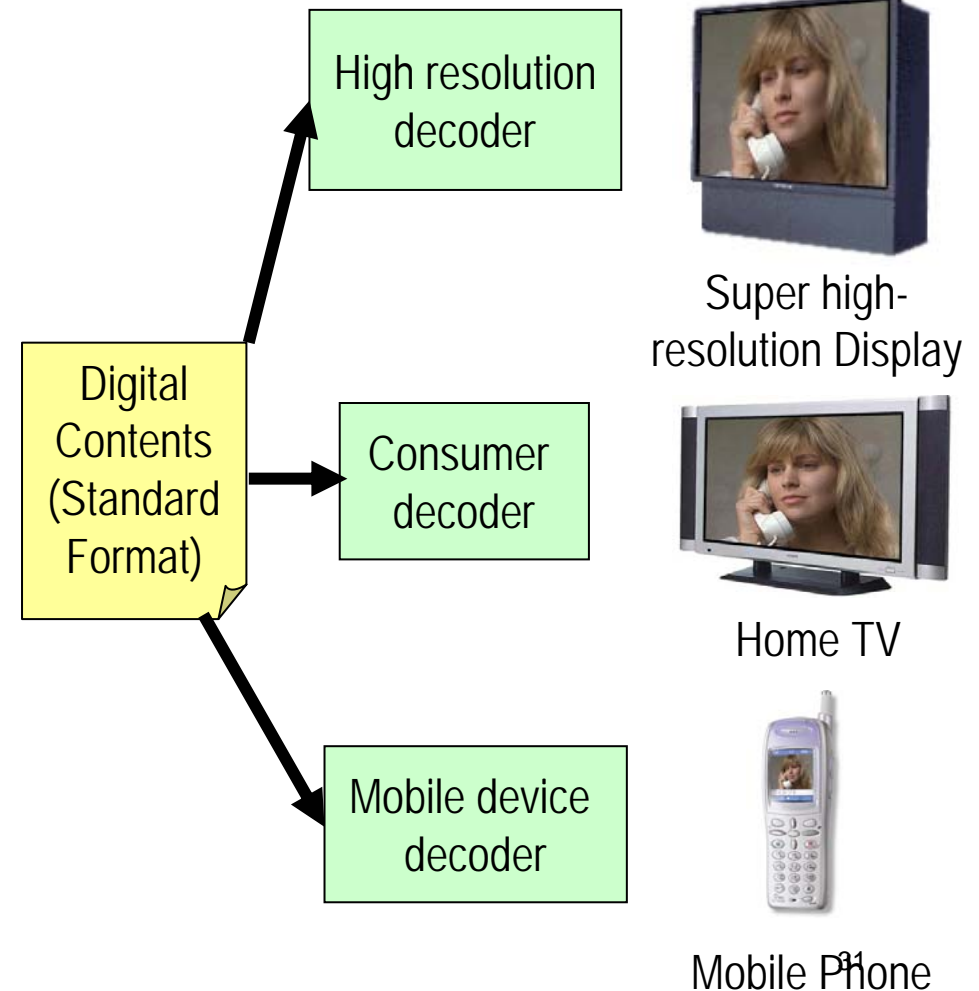
Output



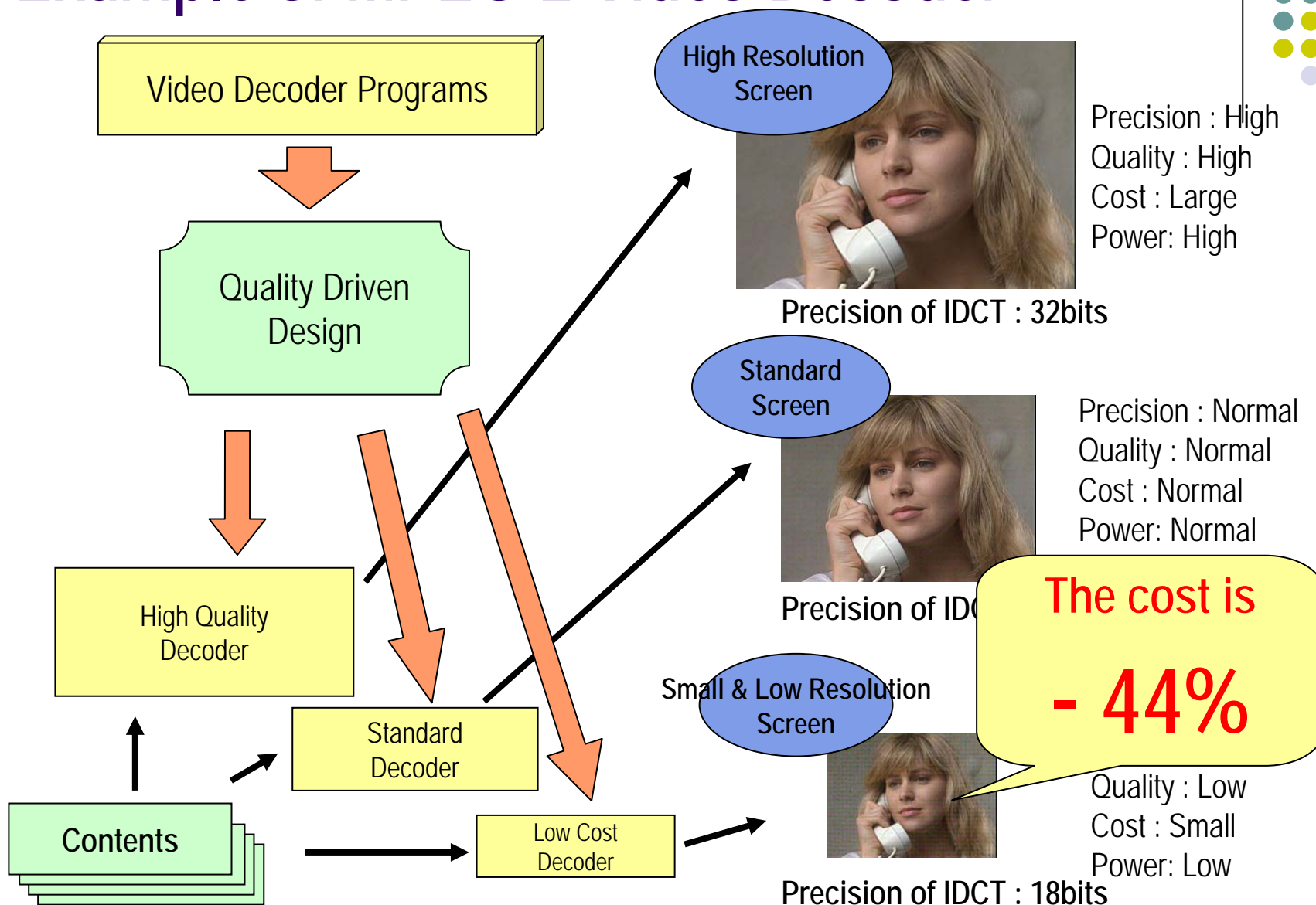
Quality-Driven Design

- *Concepts*

- Variety of output devices as human interfaces
 - Mobile devices,
 - low cost devices,
 - high-quality devices,
 - and ultra high-quality devices
- Prepare the least width of datapath for the requested accuracy of the computation



Example of MPEG-2 Video Decoder



Point 2

- Quality-Driven Design
 - Bridge between contents in standard formats and variety of output devices.
 - Kinds of the product line approach
 - From basic description and design parameter, we can get design with optimum datapath width automatically.
 - New direction of HW/SW co-design

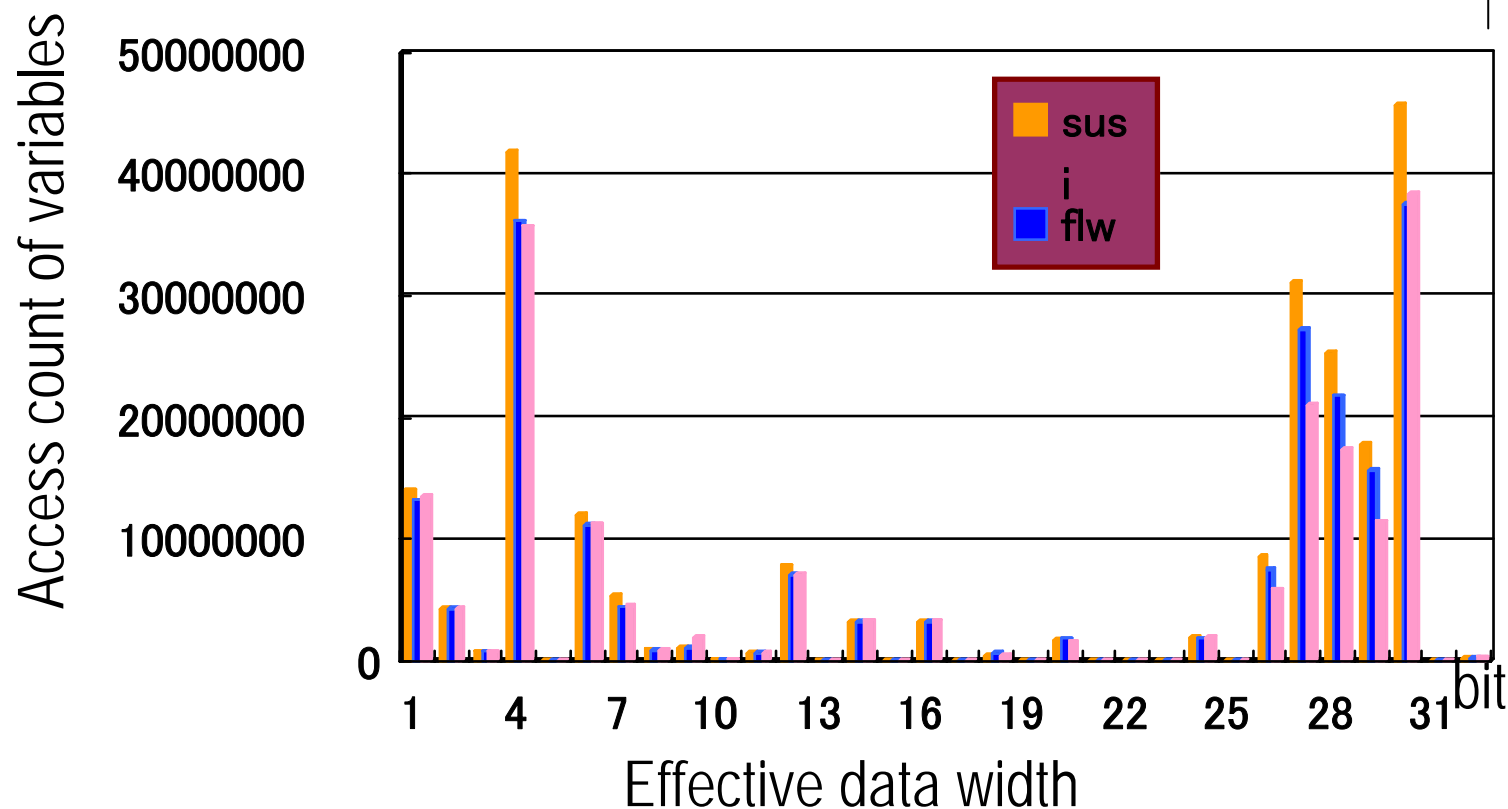


POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

- Introduction
- Datapath Width and Power Consumption
- Soft Core Processor
- **Memory Width Optimization**
- Shifted Computation
- Value-Based Clock Gating
- Conclusion

of Accesses of Variables

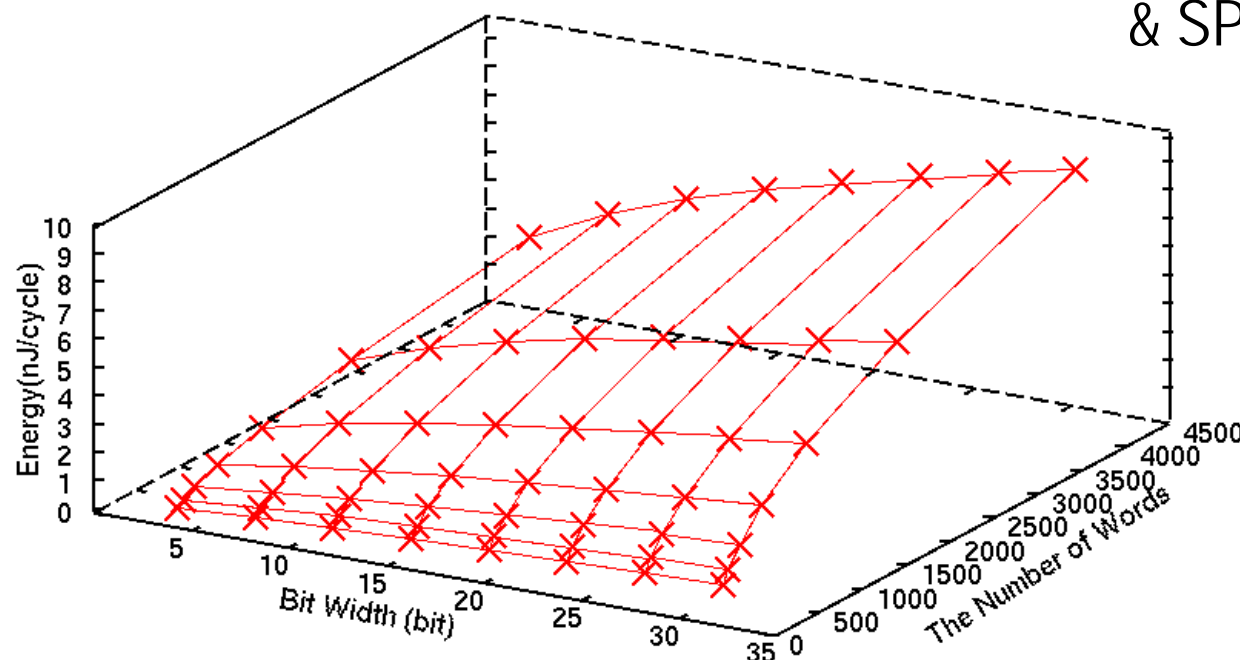
of switching depends on # of usages of data.



MPEG-2 video decoder

Memory Model

By Alliance CAD System
& SPICE simulation



Small memory
spends small
energy.

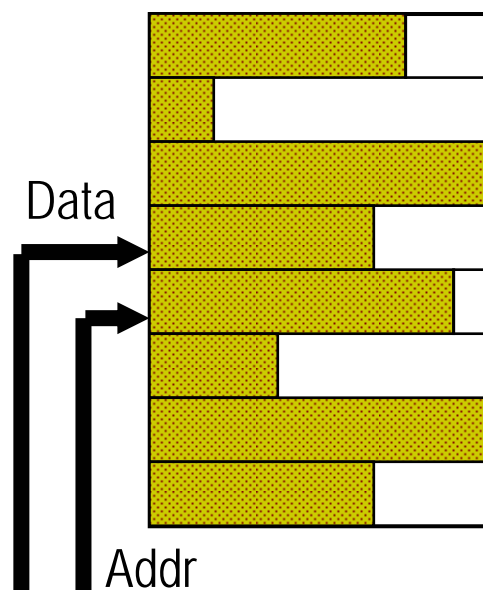
Bit width x and word count N_{word} v s. energy consumption
of read access for SRAM

$$e_r \approx 24.9 \times x \times \sqrt{N_{\text{word}}} + 56 [\text{pJ/cycle}] \quad \text{Read Access Energy}$$

$$e_w \approx 197 \times x \times \sqrt{N_{\text{word}}} + 369 [\text{pJ/cycle}] \quad \text{Write Access Energy}$$

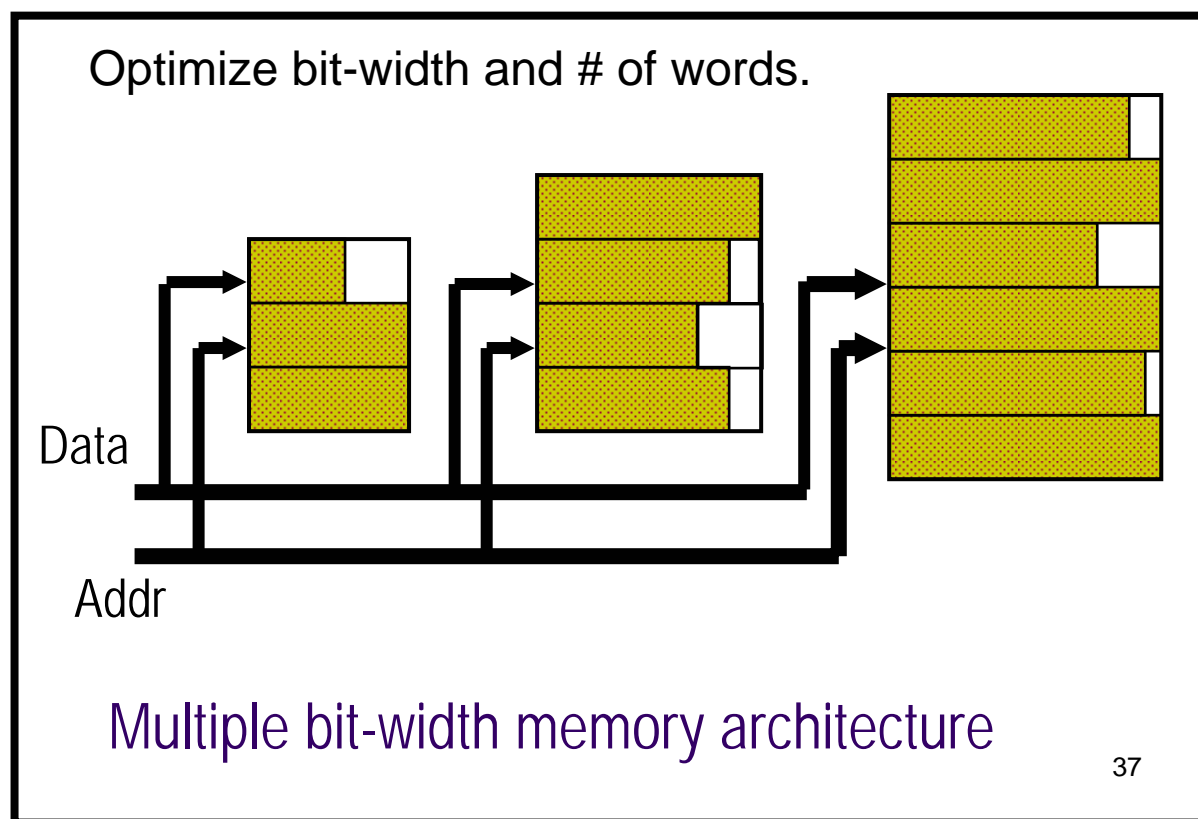
Multiple bit-width Memory Architecture

- ✧ *Data Memory is divided into multiple modules with different bit-width*
- ✧ *Assign frequently accessed data into memory module with small words*
- ✧ *Assign small size data into memory module with small bit-width*
- ✧ *Optimize the data allocation to minimize total energy consumption*



Monolithic Memory

05.10.19



Experimental Results

| Applications | Energy (J) | Memory banking Technique | | | Optimized Memory by VAbM | | |
|--------------|------------|-------------------------------------|----------|-------|--|----------|-------|
| | | Configuration | TEb(J) | Sav. | Configuration | TEm(J) | Sav. |
| Calculator | 1.27 mJ | 85 rows 154rows 533rows | 0.87 mJ | 31.5% | 85rows X 8b 154rows X 32b 533rows X 32b | 0.76 mJ | 40.2% |
| Lempel-Ziv | 1.37 | 830rows 3rows 1663rows | 0.89 | 35.0% | 830rows X 13b 3rows X 15b 1663rows X 15b | 0.69 | 49.6% |
| ADPCM | 1.63 | 20rows 16rows 86rows | 1.10 | 32.5% | 20rows X 10b 16rows X 14b 86rows X 19b | 0.80 | 50.9% |
| MPEG2AAC | 1.05 | 30rows 2374rows 4804rows | 0.39 | 62.9% | 30rows X 20b 2374rows X 32b 4804rows X 32b | 0.37 | 64.8% |
| MPEG2Video | 145.1 kJ | 26559rows 26557rows 28127rows | 120.1 kJ | 17.2% | 26559rows X 8b 26557rows X 30b 28127rows X 32b | 105.2 kJ | 27.5% |

Point 3

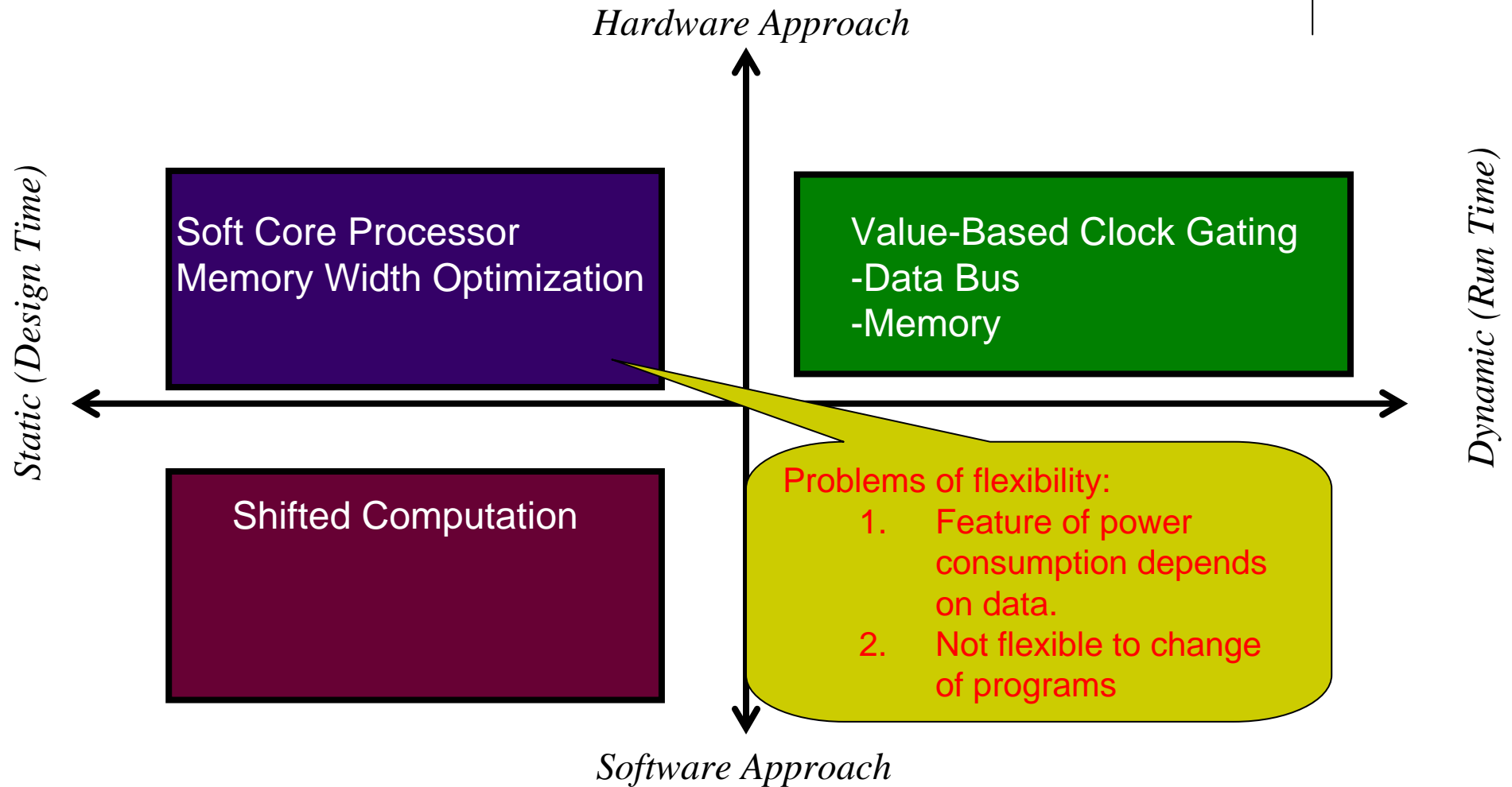
- Memory is an important target for bitwidth adjustment.
- Multiple memory banks with different bitwidth.
 - Optimization of power consumption controlling # of banks, bitwidth and # of words.
 - Up to 60% power reduction is possible.



POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

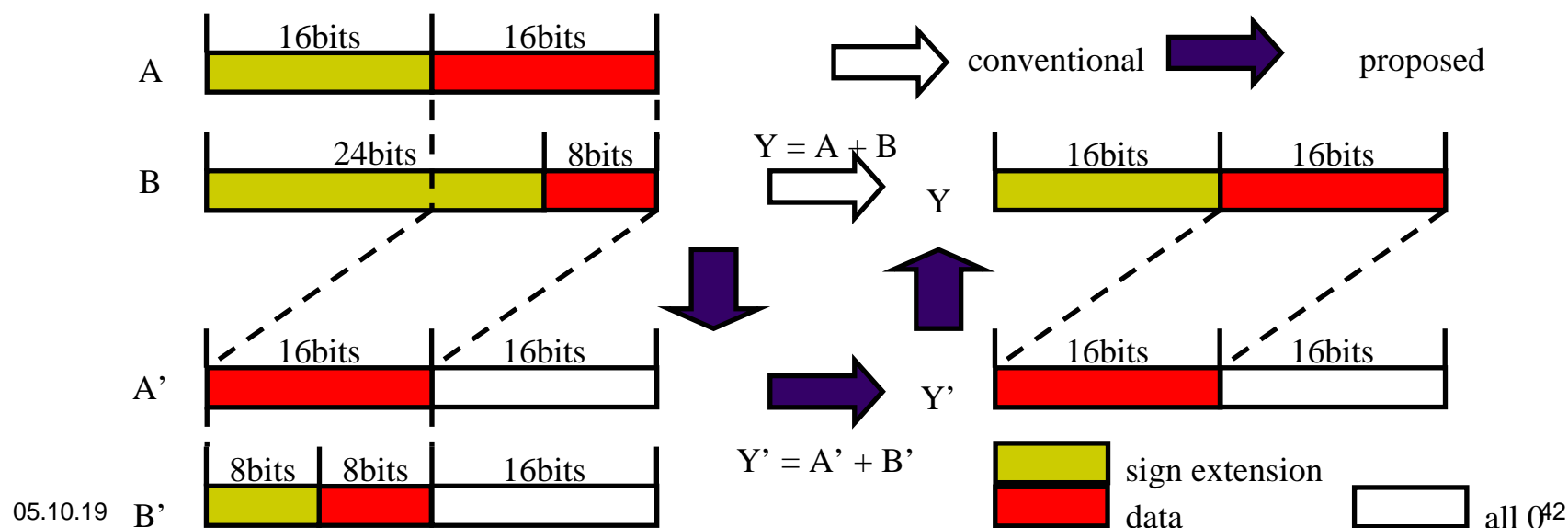
- Introduction
- Datapath Width and Power Consumption
- Soft Core Processor
- Memory Width Optimization
- Shifted Computation
- Value-Based Clock Gating
- Conclusion

Datapath Width Adjustment Techniques



Shifted Computation for Low-Precision

- *Software oriented approach*
- *Concepts*
 - For low precision computation, shifted computation is effective to reduce power consumption due to sign extension





Example of Shifted Computation for Low-Precision

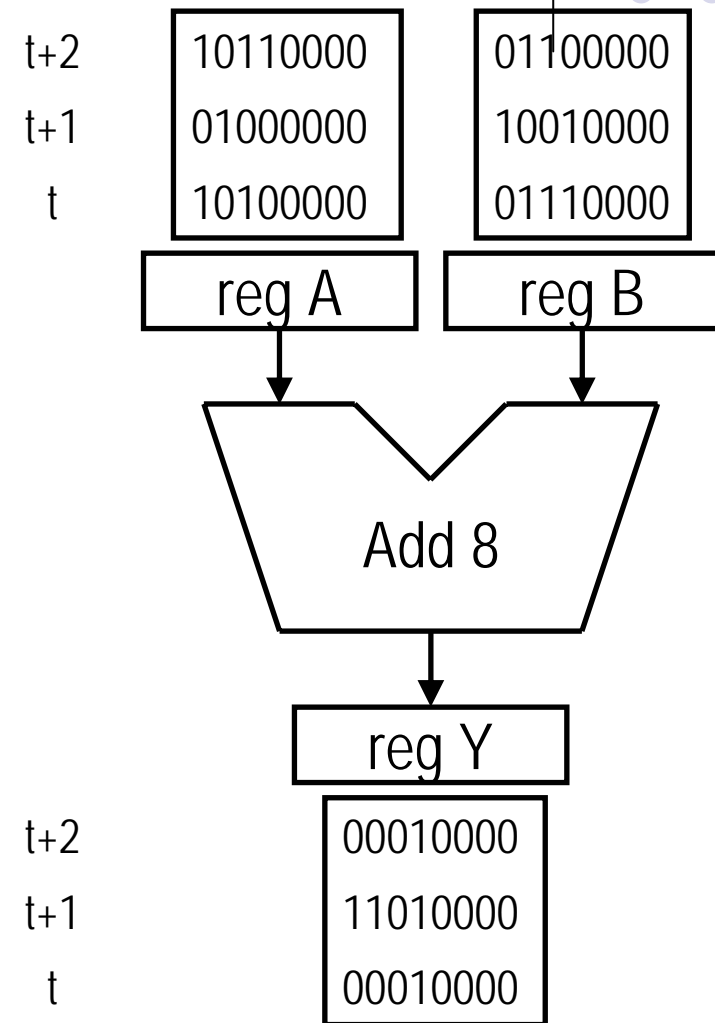
- ADPCM decoder
 - Adding shift operations based on the required bitwidth analysis
 - 32-bit CLA

Conventional
893 μ W

Proposed
551 μ W

38% saving

05.10.19



Example of the proposed tech.

Point 4

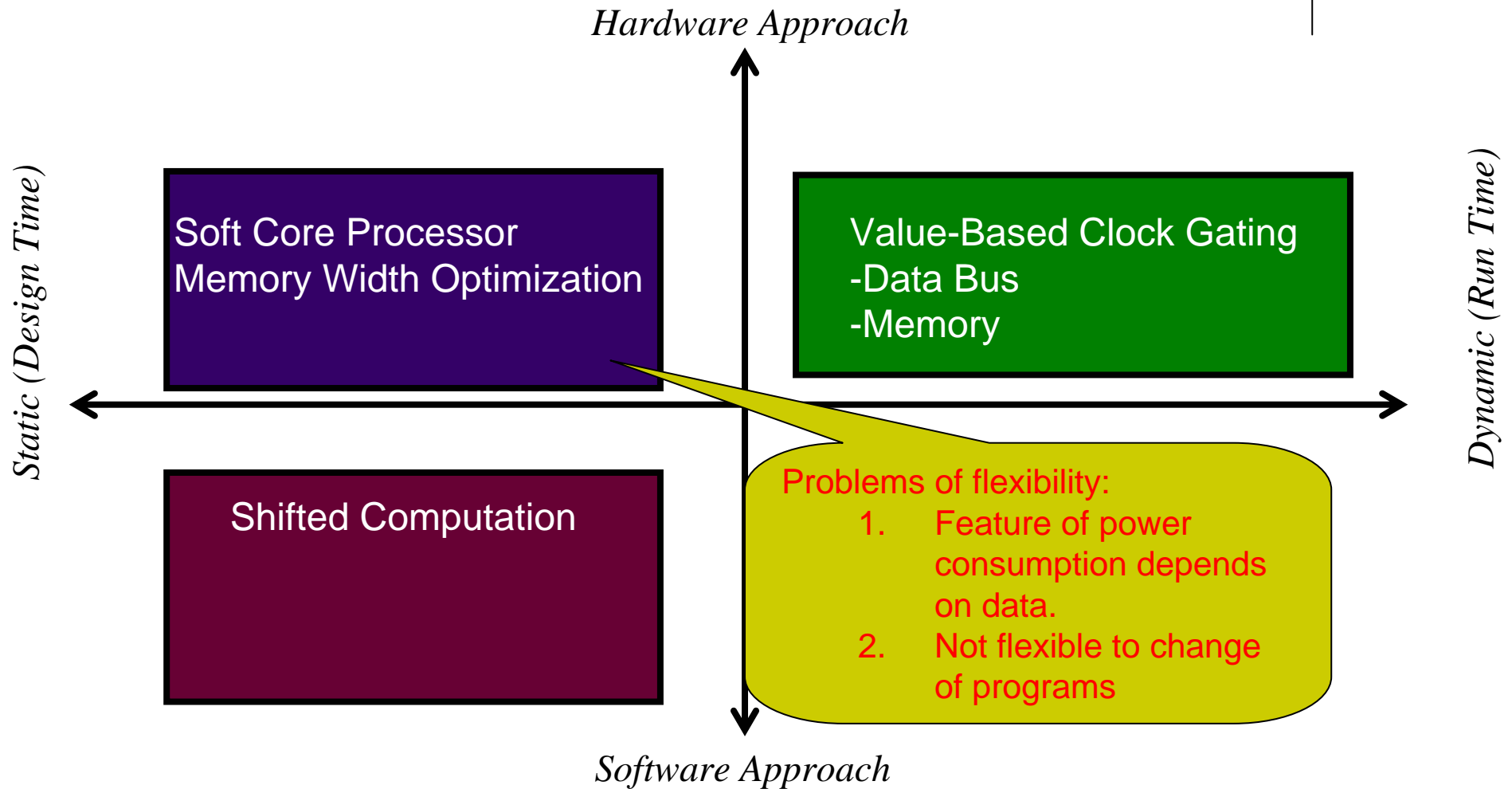
- Software oriented approach
 - No hardware change
 - SW level optimization in compilation phase
 - Overhead for shift operation in performance and power (also instruction memory).
 - On going research theme



POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

- Introduction
- Datapath Width and Power Consumption
- Soft Core Processor
- Memory Width Optimization
- Shifted Computation
- Value-Based Clock Gating
- Conclusion

Datapath Width Adjustment Techniques



Active Bits in Computation

- Active bits: the bits expressing the value of data except sign extension.
- 0000000111110001₂
- For a variable x , active bits of instances are different from each other.
 - Variable size analysis gives the worst case of the length of active bits.
 - How to identify the active bits?

How to decide Active Data Bitwidth

The case of 16 bits data width

- Method 1) remains after deleting the continuous "0" in upper bits

0000001101101101 → 10 bits

1111001101101101 → 16 bits

- Method 2) remains after deleting the continuous "1" in upper bits

0000001101101101 → 16 bits

1111001101101101 → 12 bits

- Method 3) remains after changing the continuous "0" to only one "0", or remains after changing the continuous "1" to only one "1"

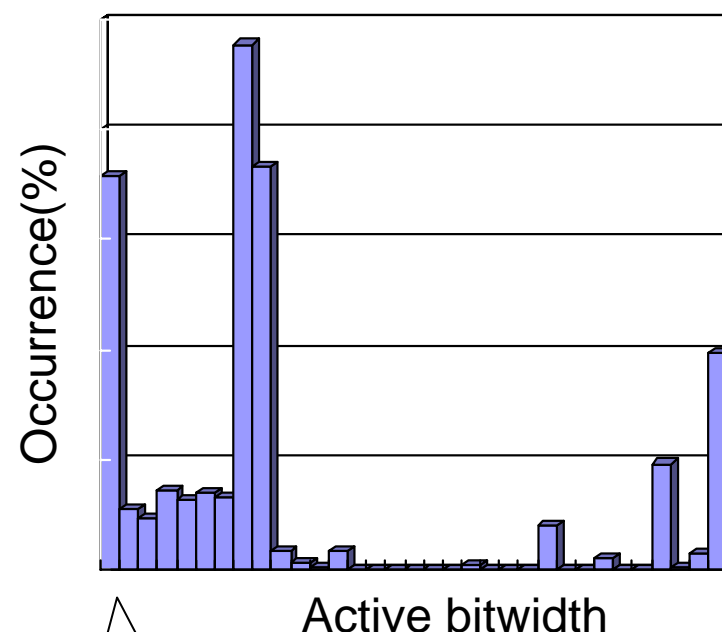
0000001101101101 → 11 bits

1111001101101101 → 13 bits

Value-Based Clock Gating

- *Concepts*

- Since small number can be expressed by using small number of bits, gating remaining redundant bits for reducing power consumption



Active bits
Active bitwidth is 9

0000000111110001₂

About 80% of active bitwidth of data is 8 or less

Example of Value-Based Clock Gating

16-bit data bus

| <i>Active bitwidth</i> | | |
|------------------------|----|--|
| 1111100101011100 | 12 | |
| 0000000111110001 | 9 | |
| 0000110101110101 | 12 | |
| 0000000000000001 | 1 | |
| 1111111111111111 | 1 | |
| 0000011010111100 | 11 | |
| 1111100110110011 | 12 | |
| 0000000111001010 | 9 | |
| <i>Data</i> | | |

of Switching 67

Original Codes

| <i>Active bitwidth</i> | <i>Onehot coding for extra bits</i> |
|------------------------|-------------------------------------|
| 1111100101011100 16 | 1000000000000000 0 |
| 1111100111110001 9 | 0000000100000000 0 |
| 1111110101110101 12 | 0000100000000000 0 |
| 1111110101110101 1 | 0000000000000001 0 |
| 1111110101110101 1 | 0000000000000001 1 |
| 1111111010111100 11 | 0000010000000000 0 |
| 1111100110110011 12 | 0000100000000000 1 |
| 1111100111001010 9 | 0000000100000000 0 |
| <i>Data</i> | <i>Extra bits</i> |

of Switching (26+16)

Encoded data using our approach

Transition reduction ratio(%): in case of onehot coding

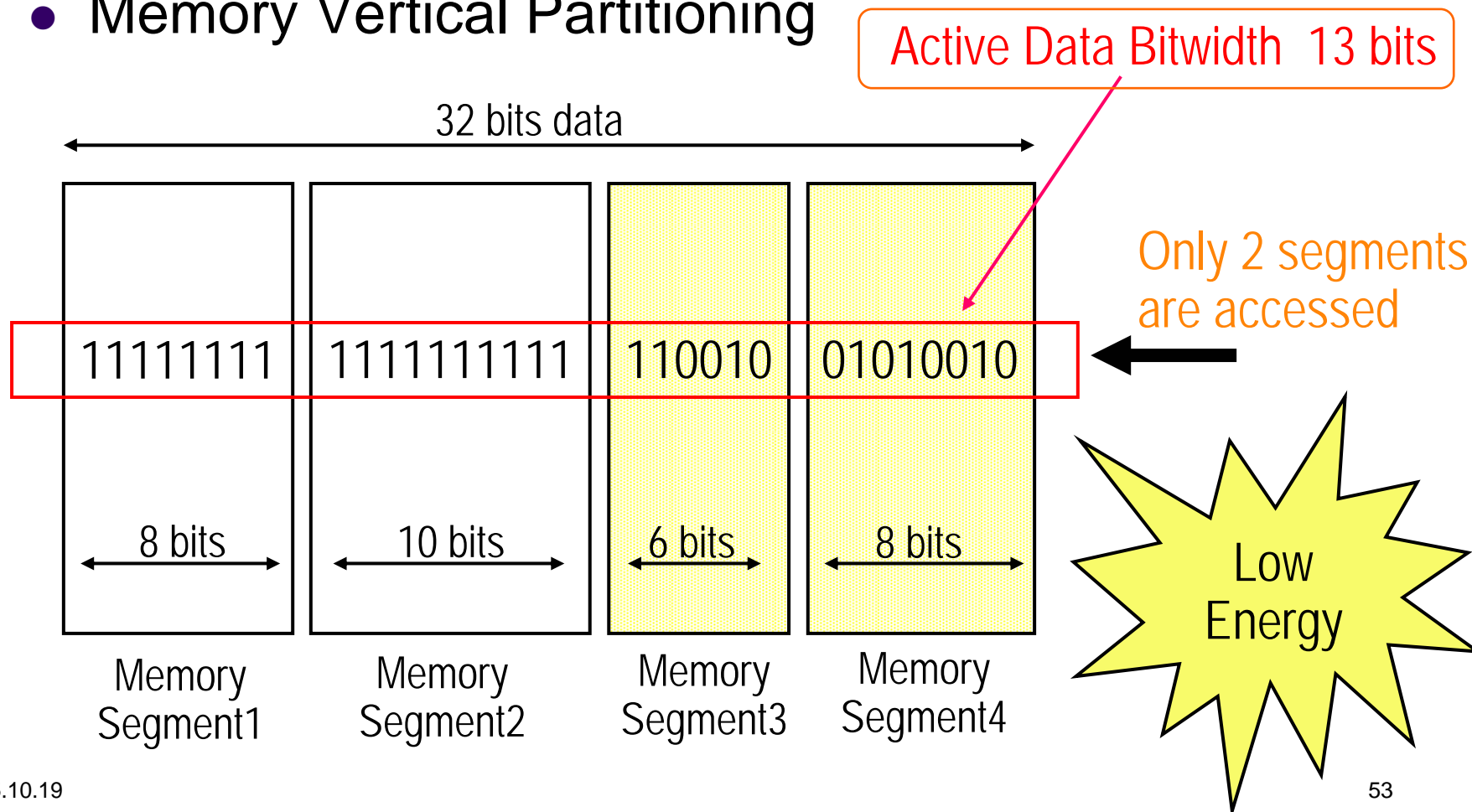


| | N _{seg} | | | | | |
|---------|------------------|-------|-------|-------|-------|-------|
| | 32 | 16 | 8 | 4 | 2 | 1 |
| mpeg2 | | | | | | |
| (116k) | -37.1 | -35.5 | -34.2 | -31.8 | -22.9 | -6.48 |
| (245k) | -45.6 | -44.1 | -42.7 | -40.3 | -27.0 | -5.19 |
| (649k) | -34.4 | -32.1 | -31.0 | -28.0 | -21.9 | -6.86 |
| go | -33.8 | -28.8 | -28.8 | -38.9 | -30.5 | -20.3 |
| gzip | -10.5 | -12.0 | -9.19 | -7.63 | -1.94 | -1.58 |
| bzip | -17.8 | -18.5 | -19.1 | -18.5 | -19.2 | -29.6 |
| perl | -14.5 | -13.8 | -11.8 | -11.5 | -3.67 | -5.25 |
| mesa | -19.0 | -17.1 | -16.1 | -13.6 | -10.8 | -15.9 |
| wupwise | -18.0 | -16.6 | -15.6 | -16.2 | -9.71 | -5.02 |
| Ave. | -22.7 | -21.6 | -20.5 | -20.9 | -14.7 | -11.8 |

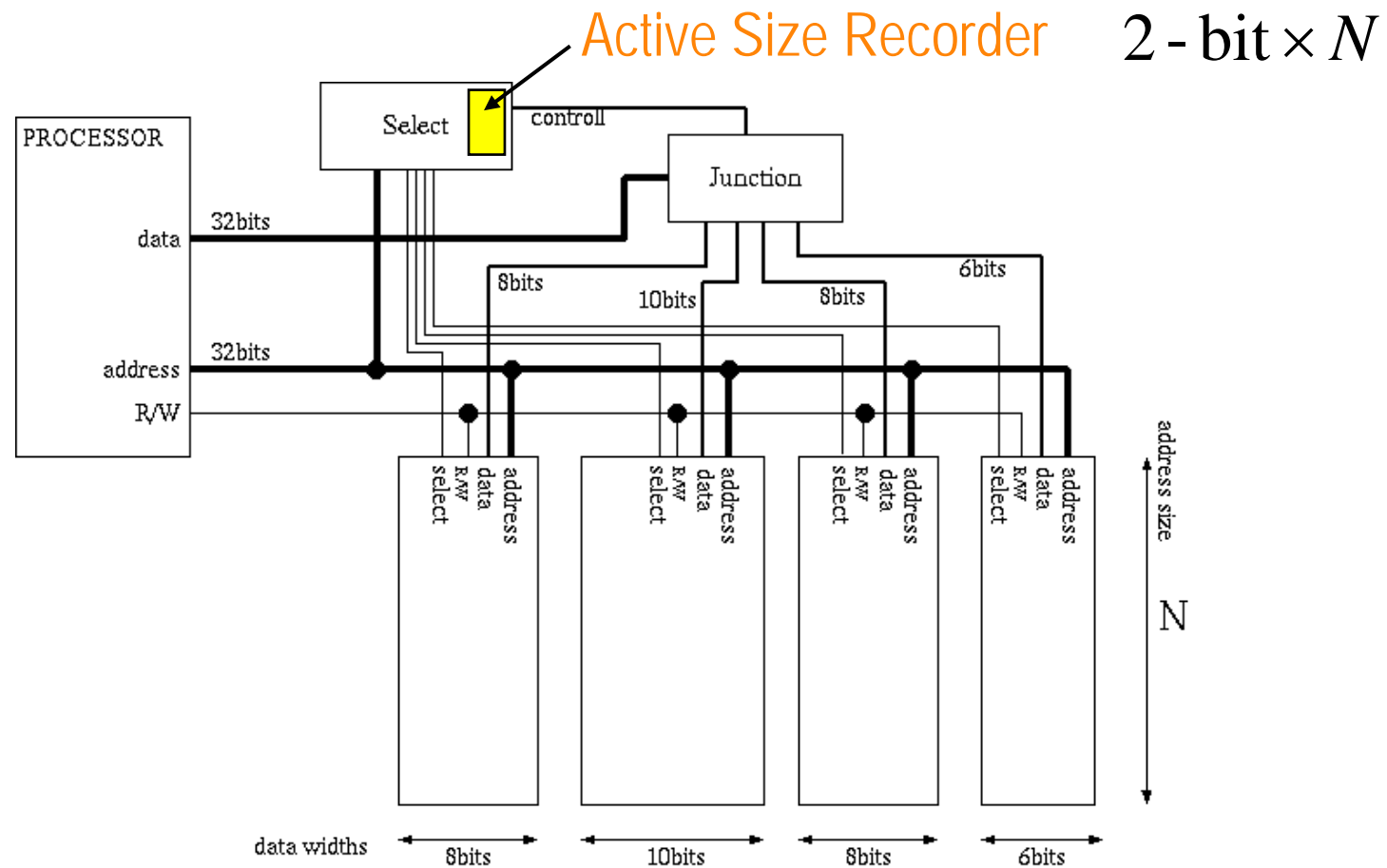
| App. | N _{seg} | | | | |
|---------|------------------|-------|--------------|--------------|--------------|
| | 32 | 16 | 8 | 4 | 2 |
| mpeg | | | | | |
| (116k) | -30.0 | -35.4 | -37.9 | -38.6 | -26.9 |
| (245k) | -40.7 | -44.8 | -46.5 | -45.3 | -30.9 |
| (649k) | -26.7 | -31.3 | -34.4 | -35.8 | -26.0 |
| go | -23.7 | -28.8 | -39.0 | -49.1 | -20.3 |
| gzip | -8.83 | -11.6 | -12.7 | -12.2 | -8.13 |
| bzip2 | -7.19 | -13.1 | -19.1 | -25.6 | -31.1 |
| perl | -13.3 | -10.8 | -14.2 | -16.7 | -12.0 |
| mesa | -6.75 | -9.30 | -14.5 | -20.0 | -22.1 |
| wupwise | -12.3 | -13.9 | -17.7 | -18.7 | -15.2 |
| Ave. | -16.1 | -18.9 | -23.1 | -26.8 | -20.0 |

Applicaton to Memory

- Memory Vertical Partitioning



Memory Partitioning Example



Experimental Results

| Mpeg2play N _{word} 302834 | Memory | | Active Size Recorder | | Total Energy |
|---------------------------------------|--------|-------|----------------------|-------|--------------|
| | Read | Write | Read | Write | |
| Our Technique | 0.75J | 0.59J | 0.14J | 0.74J | 2.22J |
| Monolithic Memory | 2.3J | 11.8J | | | 14.1J |
| Memory Banking | 1.08J | 5.43J | | | 6.51J |

| IJPEG N _{word} 917987 | Memory | | Active Size Recorder | | Total Energy |
|-----------------------------------|--------|--------|----------------------|-------|--------------|
| | Read | Write | Read | Write | |
| Our Technique | 3.16J | 4.36J | 0.37J | 0.76J | 8.65J |
| Monolithic Memory | 5.94J | 12.17J | | | 18.11J |
| Memory Banking | 4.58J | 5.28J | | | 9.86J |

$$N_{seg} = 4$$

$$b_1 = 8$$

$$b_2 = 8$$

$$b_3 = 8$$

$$b_4 = 8$$

Point 5

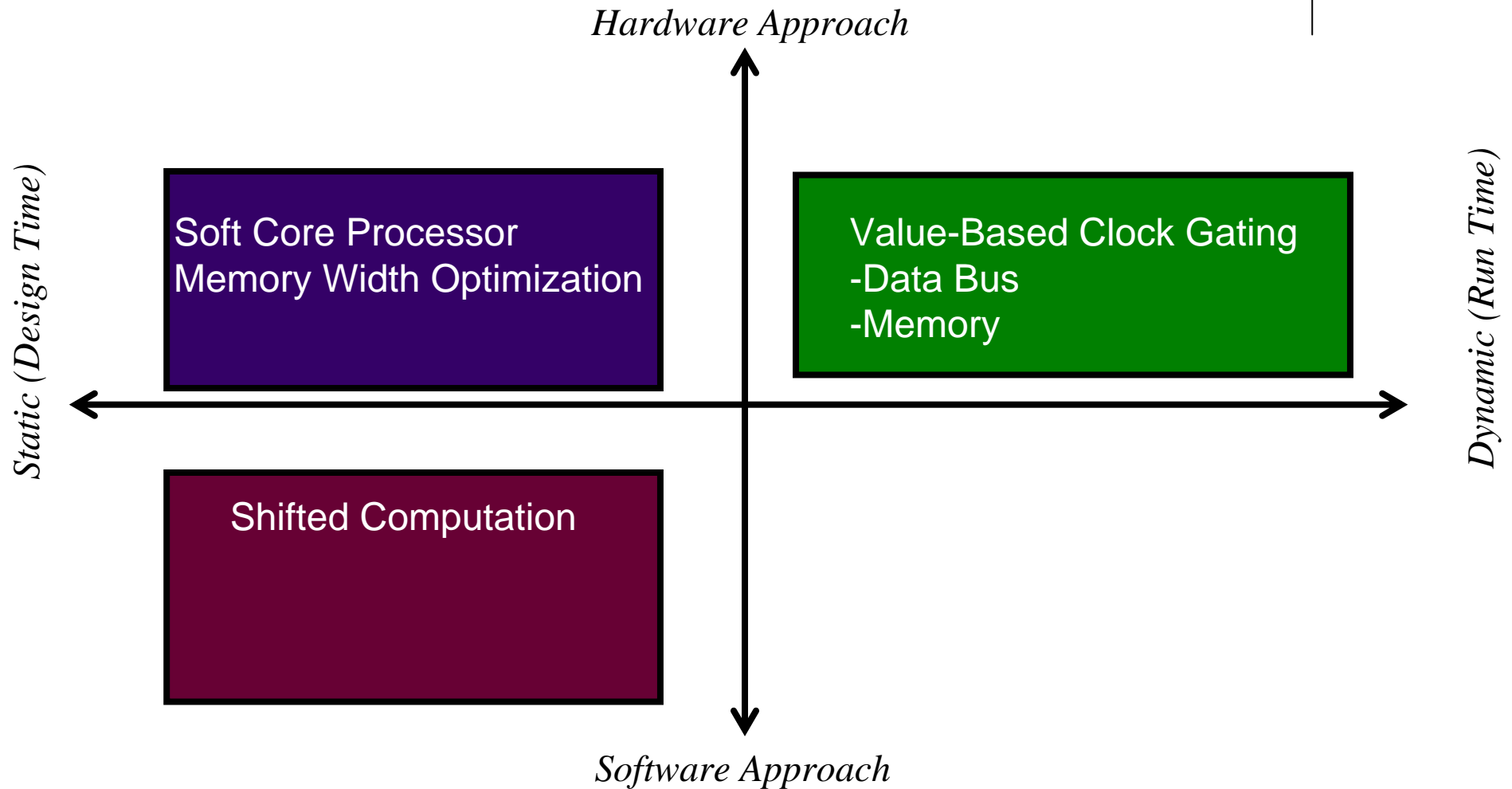
- Active data bitwidth is a dynamic approach for datapath adjustment.
- Overheads
 - How to recognize active bits.
 - How to keep the information of the active bits.
 - How to control switching.
- No effects in SW level.
- Applicable for Logic, Processors and Memories.



POWER OPTIMIZATION BY DATAPATH WIDTH ADJUSTMENT

- Introduction
- Datapath Width and Power Consumption
- Soft Core Processor
- Memory Width Optimization
- Shifted Computation
- Value-Based Clock Gating
- Conclusion

Datapath Width Adjustment Techniques



Datapath Width Adjustment

- Datapath adjustment is an important method for power and/or energy reduction.
- Various approaches are possible to apply for logic circuits, processors, memories and software.
- It is a new direction of reconstruction of compiler and processor architecture technology.
- SoC requires clear and simple principles for system level optimization. Through the researches on the datapath adjustment, we can find a hints of system level optimization approaches.



References

- General

- Takanori Okuma, Tohru Ishihara, and Hiroto Yasuura, "Software Energy Reduction Techniques for Variable Voltage Processors", IEEE Design & Test of Computers March-April 2001, Vol.18, No.2, pp.31-41, Apr. 2001.

- Soft Core Processor

- Hiroto Yasuura and Hiroyuki Tomiyama, "Power Optimization by Datapath Width Adjustment", In Massoud Pedram and Jan M. Rabaey, editors, Power Aware Design Methodologies, chapter 7, pp.181-199, Kluwer Academic Publishers, Jun. 2002.
- Yun Cao and Hiroto Yasuura, "A System-level Energy Minimization Approach Using Datapath Width Optimization", IEICE Technical Report, VLD2000-142, ICD2000-218(2001-03), pp.49-54, Mar. 2001.

- Quality Driven Design

- Yun Cao and Hiroto Yasuura, "Video Quality Modeling for Quality-driven Design", The 10th Workshop on System And System Integration of Mixed Technologies(SASIMI 2001), pp.86-92, Oct. 2001.
- Yun Cao and Hiroto Yasuura, "Quality-Driven Design for Video Applications", IEICE Transactions on Fundamentals of Electronics, Vol.E85-A, No.12, pp.2568-2576, Dec. 2002.

- Memory Width Optimization

- Yun Cao and Hiroto Yasuura, "A Low-Energy Memory Design Technique Based on Variable Analysis for Application-Specific Systems", ACM the 12th Great Lakes Symposium on VLSI (GLSVLSI2002), Apr. 2002.
- Yun Cao and Hiroto Yasuura, "Memory Organization for Low-Energy Processor-Based Application-Specific Systems", IEICE Transactions on Electron, Vol.J83-C, No.8, pp.1616-1624, Aug. 2002.

- Value-Based Clock Gating

- Masanori Muroyama, Akihiko Hyodo, Takanori Okuma, and Hiroto Yasuura, "A Power Reduction Scheme for Data Buses by Dynamic Detection of Active Bits", Proceedings of Euromicro Symposium on Digital System Design -Architectures, Methods and Tools-(DSD2003), pp.408-415, Sep. 2003.
- Masanori Muroyama, Akihiko Hyodo, Takanori Okuma, and Hiroto Yasuura, "A Power Reduction Scheme for Data Buses by Dynamic Detection of Active Bits", IEICE Transactions on Electronics, Vol.E87-C, No.4, pp.598-605, Apr. 2004.
- Takanori Okuma, Yun Cao, Masanori Muroyama, and Hiroto Yasuura, "Reducing Access Energy of On-Chip Data Memory Considering Active Data Bitwidth", Proc. of International Symposium on Low Power Electronics and Design(ISLPED'02), pp.88-91, Aug. 2002.

Thank you