Myricom's FPGA based Approach to ATR/SLD

ARPA/ACS-PI-Mtg, Nov'97

Wen-King Su, Young Cho,
Ruth Sivilotti, Danny Cohen,
and Brian K. Bray (of SNL)

Myricom, Inc.
325 N. Santa Anita Ave, Arcadia, CA 91006
818-821-5555 <Cohen@Myri.com> Fx: 818-821-5316
http://www.myri.com

Overview


The Problem


Architecture


The Computation


The Implementation


Conclusion

The ATR/SLD Task

Given a set of templates, organized by:
{ class | object | depression | orientation }
Example: { missiles -> scud -> steep -> 75^{o} }.

Each object has many templates (at least 3*72).

The templates are static (i.e., fixed for a mission), but this may change for model-based algorithms. This approach can handle both cases.

Task: Match templates to given images.

The best K valid matches suggest probable targets in the image. (K=1,2,...)

Arrays

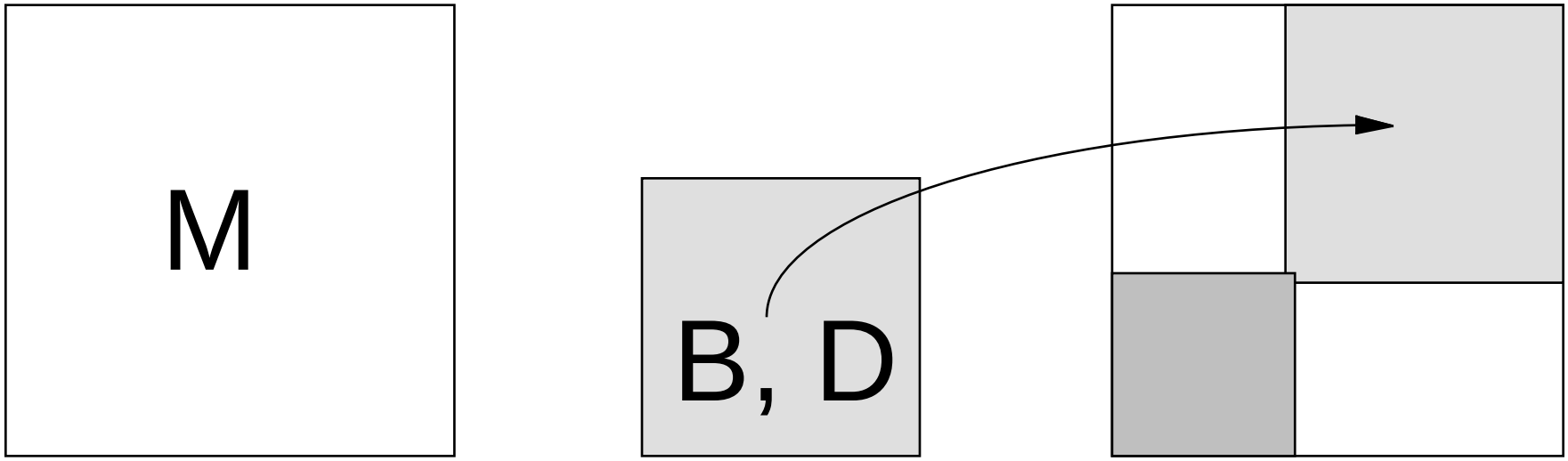


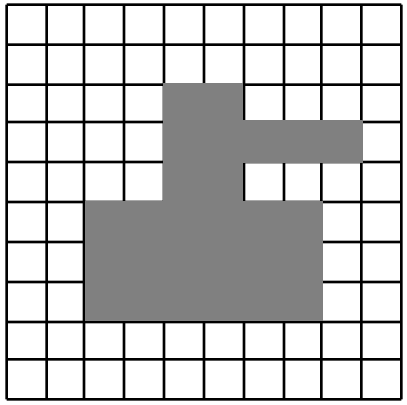
Image (52x52)

Template (32x32)

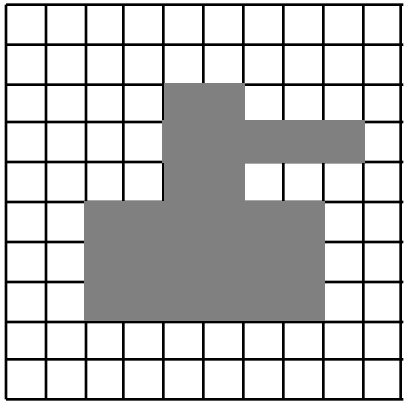
Search Area (21x21)

52 - 32 + 1 = 21

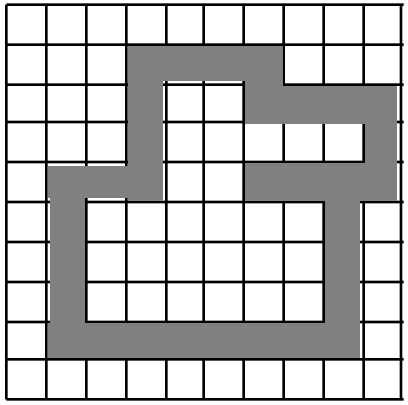
Templates



Target



Bright template $B_{i,j}$



Surround template $D_{i,j}$

Data Organization

A Template is 2 arrays of 32x32x1bit, and a few parameters (Bias,Θ_{min},Θ_{max},BS_{min},DS_{min}), ~270B

An Image is an array of 64x64x1B, ~4KB

A Match-Task is an Image and IDs of templates (probably in two intervals).

A Hit is a degree of match between a template and an image.

A Hit-Report includes image-ID, hit position, template-ID, and the hit-value.

Division of Labor

The SLD host processes images, determines areas of interest, identifies FOAs and issues Match-Tasks for images and sets of templates.

The SLD processors asynchronously return Hit-Reports, each with an image-ID, hit position, template-ID, and the best K valid hit-values (K=1,2,...).

Architecture 1

Each of the N processing nodes holds all the templates, and handles an entire Match-Task by itself.

At startup each node gets a few Match-Tasks. Thereafter, when a node returns a Hit-Report, it gets a new Match-Task.

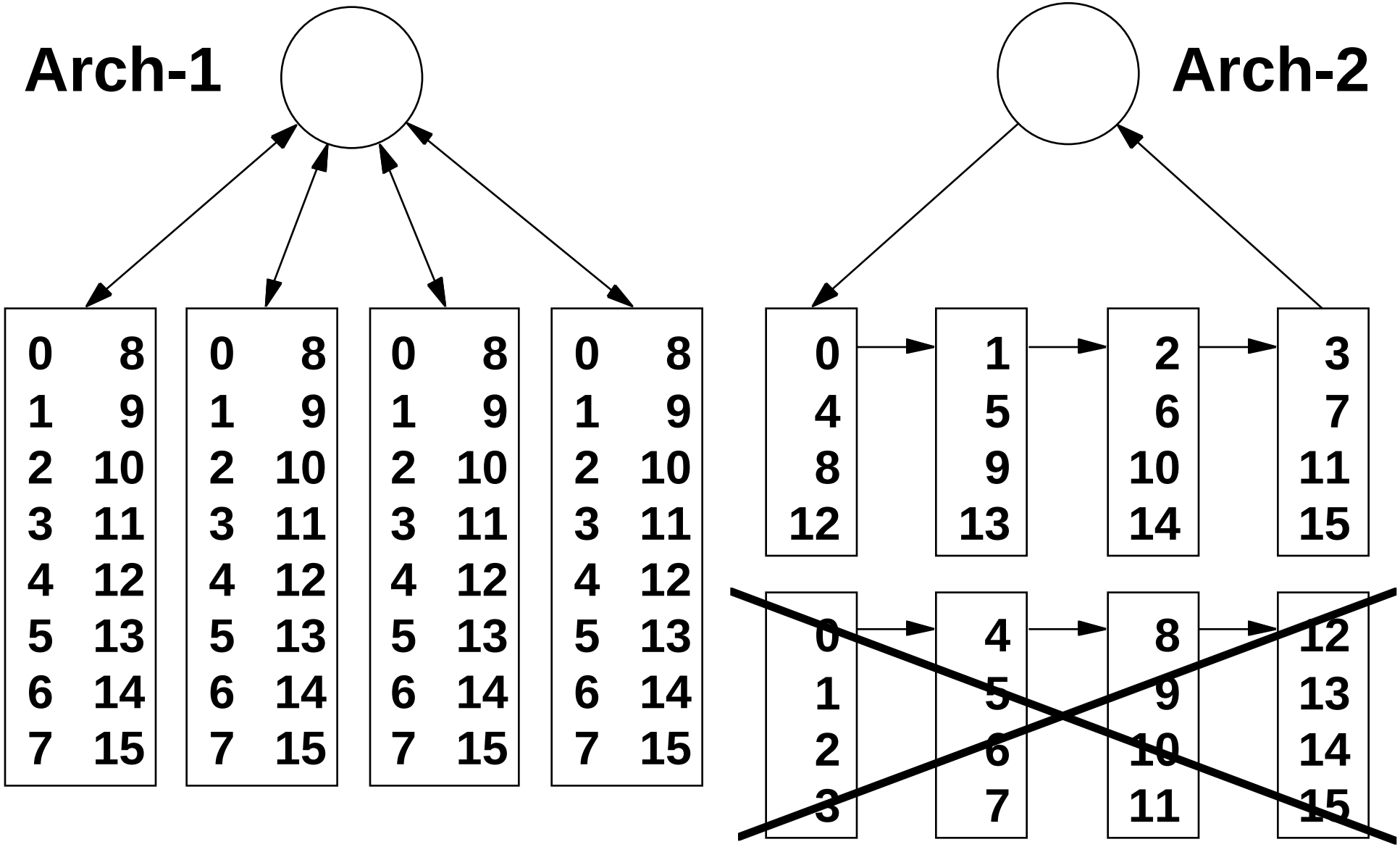
Architecture 2

Each of the N processing nodes holds only 1/N of the templates. Each matches the image with all of its own templates, and forwards to the next node a partial ("up-to-here") Hit-Report and the Match-Task.

All node participate in each Match-Task.

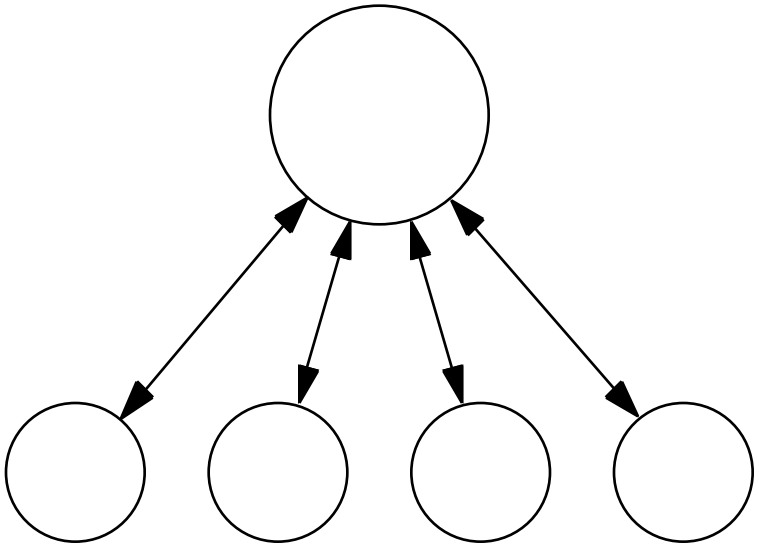
All data flows in one stream. All tasks are given to the first node, and all Hit-Reports arrive to the host from the last node.

Scaleability

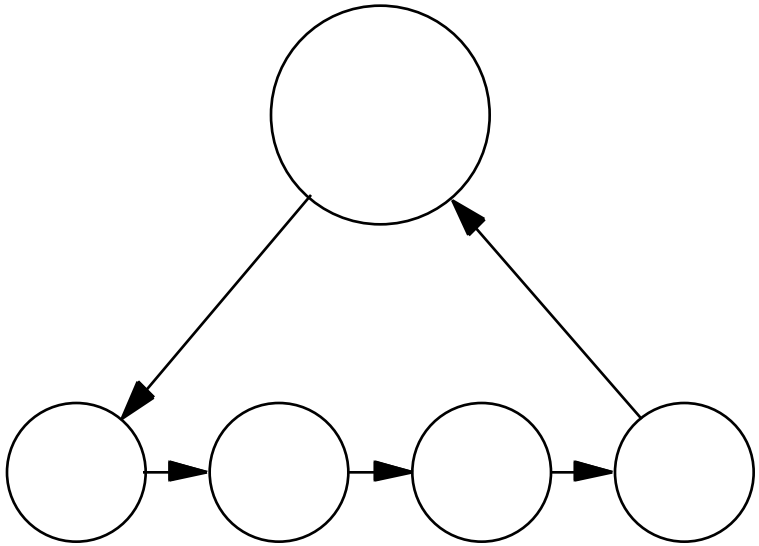


Comparison

Architecture 1

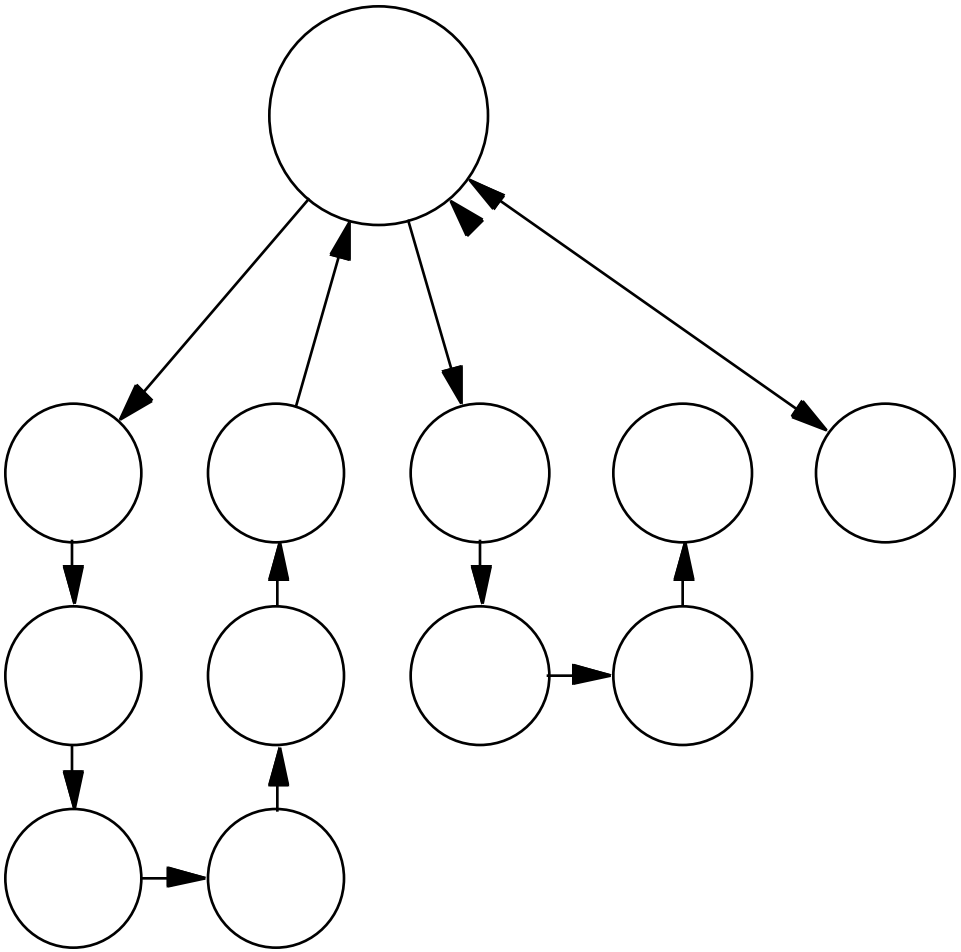


Architecture 2



Scaleability	<<	Scaleability
Fault Tolerance	>	Fault Tolerance
Load Balancing	>	Load Balancing

Architecture 3 (hybrid)



Info Flow in Each Node

The LANai receives a Match-Task and prepares pointers for the FPGA to perform matches, and to return match values for each position.

The LANai reports the best K valid matches in the Hit-Report, which is sent [Arch1] to the host, or [Arch2] to the next node (with the Match-Task).

The Computation

$$S_{i,j} = \sum_{a=0}^{31} \sum_{b=0}^{31} B_{a,b} M_{a+i,b+j}$$

$$\Theta_{i,j} = (S_{i,j} / B_{count}) - \text{Bias}$$

$$BS_{i,j} = \sum_{a=0}^{31} \sum_{b=0}^{31} B_{a,b} (M_{a+i,b+j} \geqslant \Theta_{i,j})$$

$$DS_{i,j} = \sum_{a=0}^{31} \sum_{b=0}^{31} D_{a,b} (M_{a+i,b+j} < \Theta_{i,j})$$

The Constraints

A hit is valid if it satisfies:

$$\Theta_{i,j} < \Theta_{max}$$

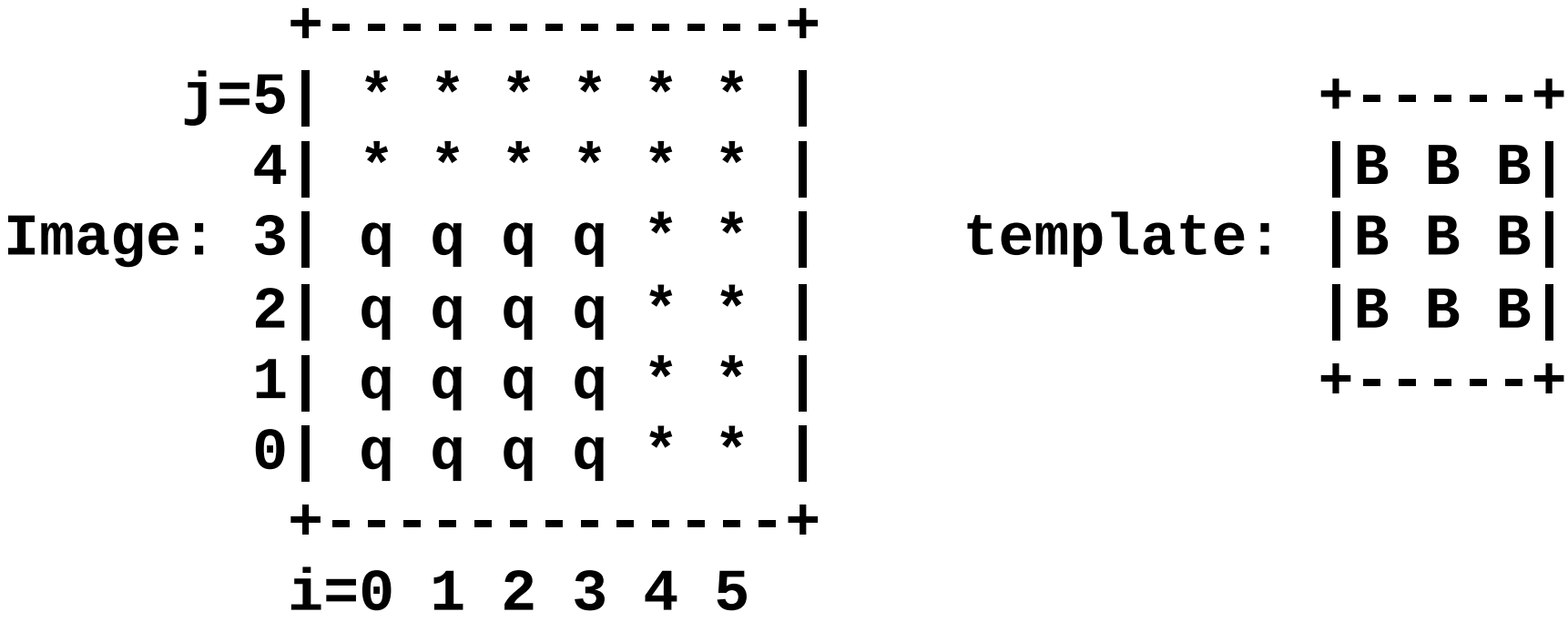
$$\Theta_{i,j} > \Theta_{min}$$

$$BS_{i,j} > BS_{min}$$

$$DS_{i,j} > DS_{min}$$

Example-1

Images of size 6x6: $M_{i,j}$ for i={0-5}, j={0-5}
Templates of 3x3: $B_{i,j}$ for i={0-2}, j={0-2}
Search area 4x4: $Q_{i,j}$ for i={0-3}, j={0-3}



Example-2

$$Q(i,j) = \sum_{a=0}^{2} \sum_{b=0}^{2} B(a,b) * M(a+i,b+j)$$

```
Qi,j = B00*M(i,  j) + B01*M(i,  j+1) + B02*M(i  ,j+2) +
       B10*M(i+1,j) + B11*M(i+1,j+1) + B21*M(i+1,j+2) +
       B20*M(I+2,j) + B21*M(I+2,j+1) + B22*M(i+2,j+2)
```

$Q_{i,j}$ is computed by the unit U_j for j={0-3}.

4 parallel units.

Example-3

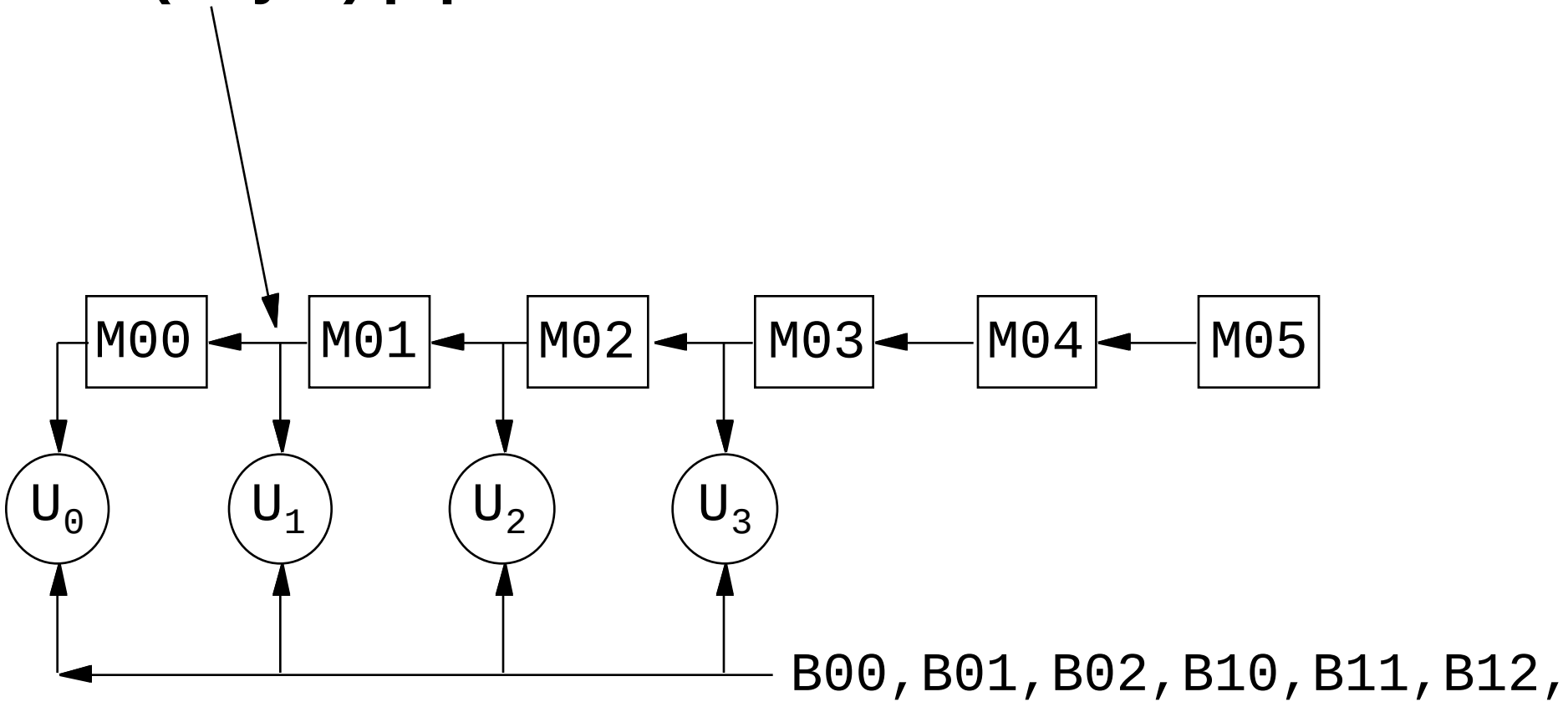
```
U0: q00=B00*M00+B01*M01+B02*M02+B10*M10+B11*M11+B12*M12+B20*M20+B21*M21+B22*M22
U1: q01=B00*M01+B01*M02+B02*M03+B10*M11+B11*M12+B12*M13+B20*M21+B21*M22+B22*M23
U2: q02=B00*M02+B01*M03+B02*M04+B10*M12+B11*M13+B12*M14+B20*M22+B21*M23+B22*M24
U3: q03=B00*M03+B01*M04+B02*M05+B10*M13+B11*M14+B12*M15+B20*M23+B21*M24+B22*M25
```

	t=0	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8
	B00	B01	B02	B10	B11	B12	B20	B21	B22
U3:	M03	M04	M05	M13	M14	M15	M23	M24	M25
U2:	M02	M03	M04	M12	M13	M14	M22	M23	M24
U1:	M01	M02	M03	M11	M12	M13	M21	M22	M23
U0:	M00	M01	M02	M10	M11	M12	M20	M21	M22

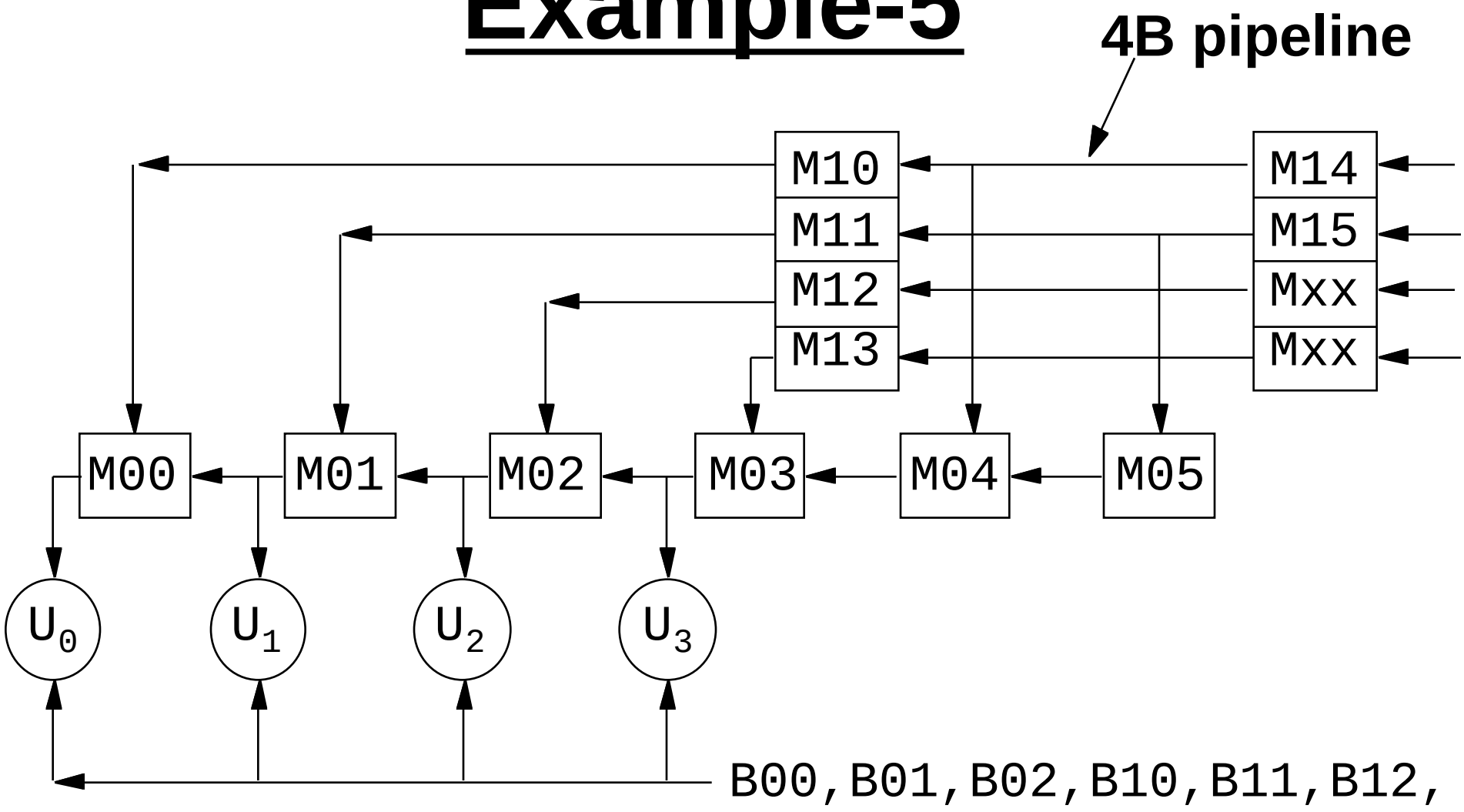
$Q_{i,j}$ is computed with: i in-time (sequentially)
j in-place (in parallel)

Example-4

Pixel (1Byte) pipeline

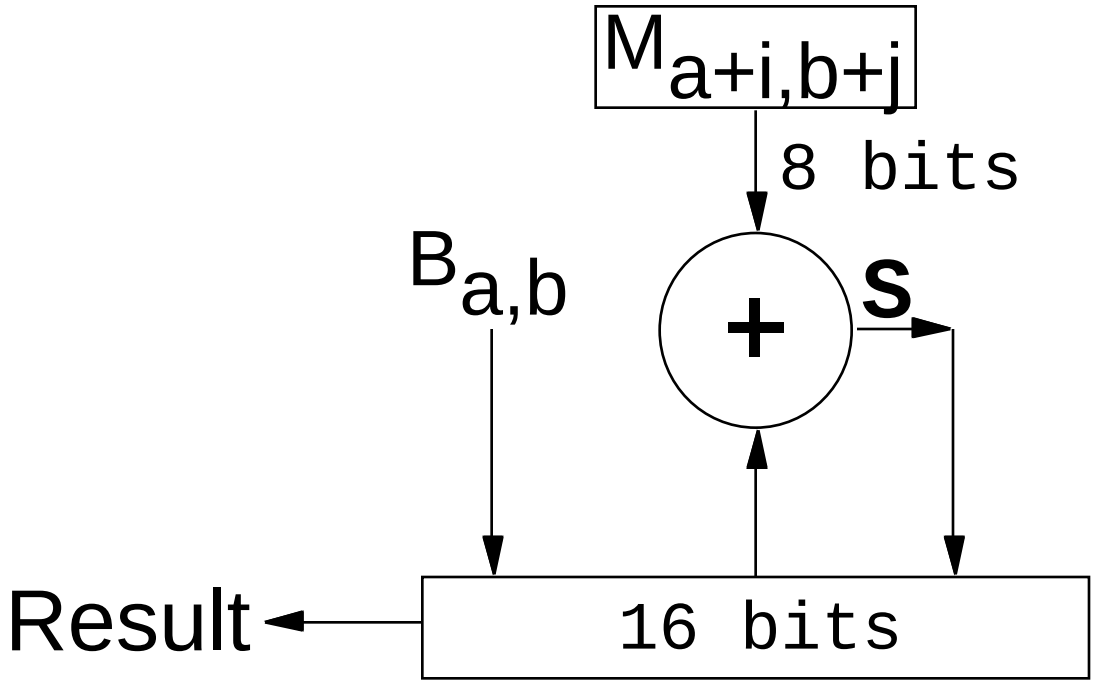


Example-5


4B pipeline
M10
M11
M12
M13
M14
M15
Mxx
Mxx
M00
M01
M02
M03
M04
M05
U0
U1
U2
U3
B00,B01,B02,B10,B11,B12,

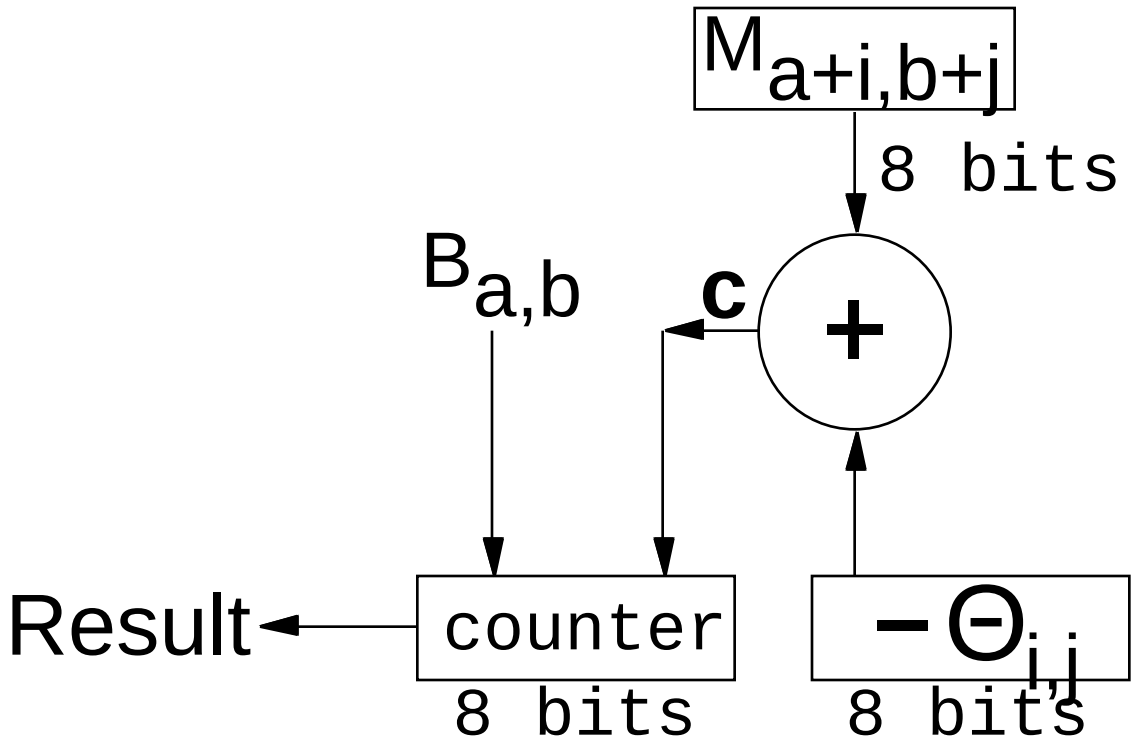
U_j: Computing $S_{i,j}$

$$S_{i,j} = \sum_{a=0}^{31} \sum_{b=0}^{31} B_{a,b} M_{a+i,b+j}$$



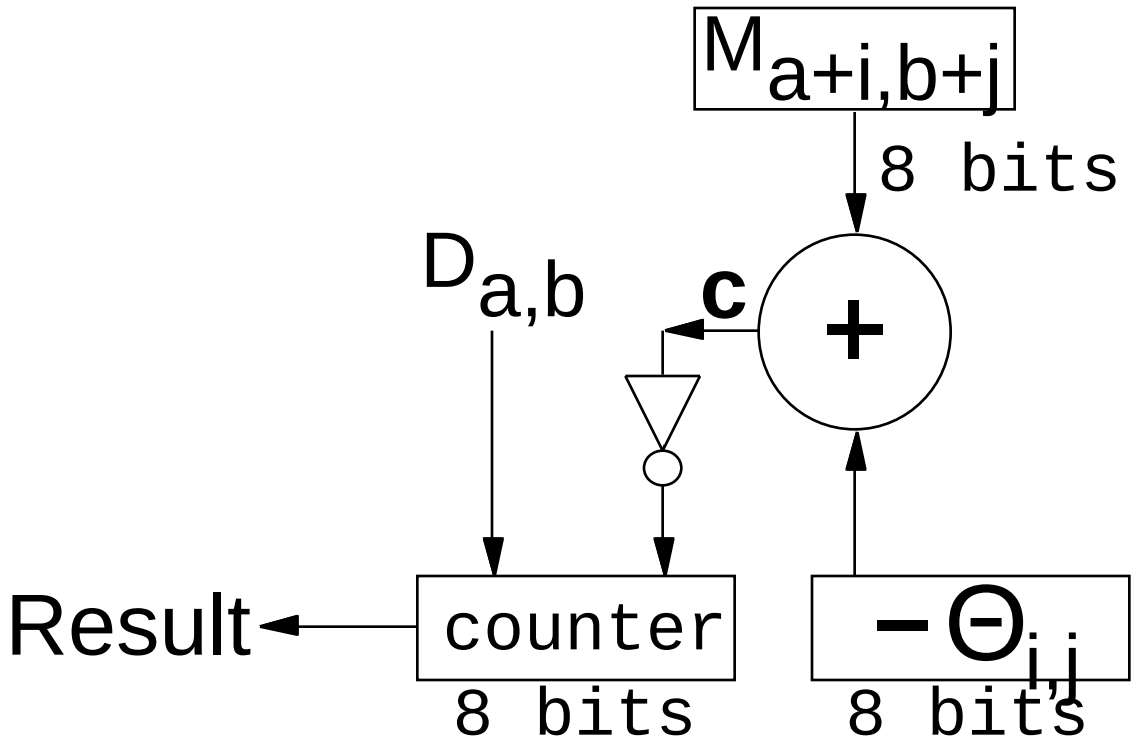
U_j: Computing $BS_{i,j}$

$$BS_{i,j} = \sum_{a=0}^{31} \sum_{b=0}^{31} B_{a,b} \left(M_{a+i,b+j} \geq \Theta_{i,j}\right)$$

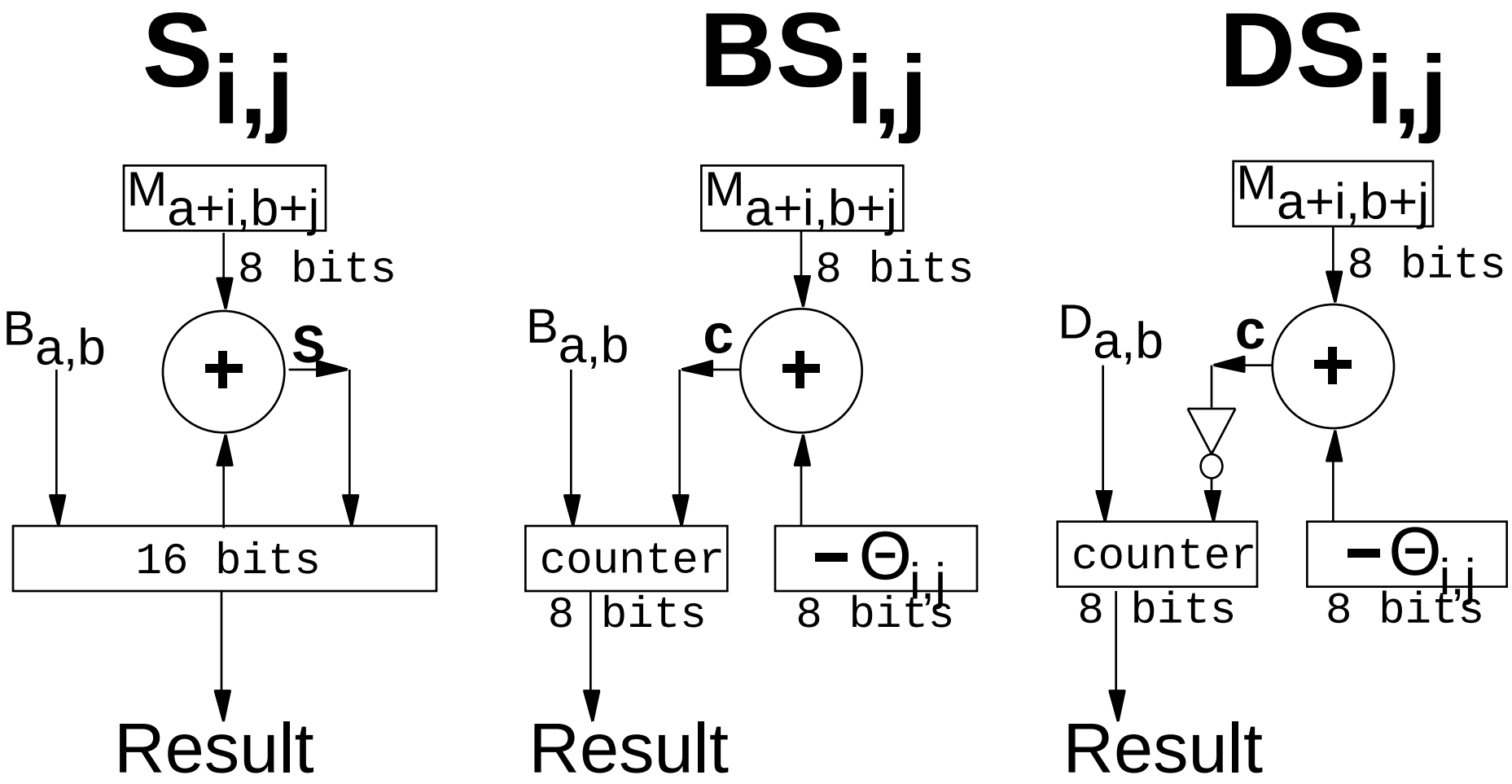


U_j: Computing $DS_{i,j}$

$$DS_{i,j} = \sum_{a=0}^{31} \sum_{b=0}^{31} D_{a,b} \left(M_{a+i,b+j} < \Theta_{i,j}\right)$$

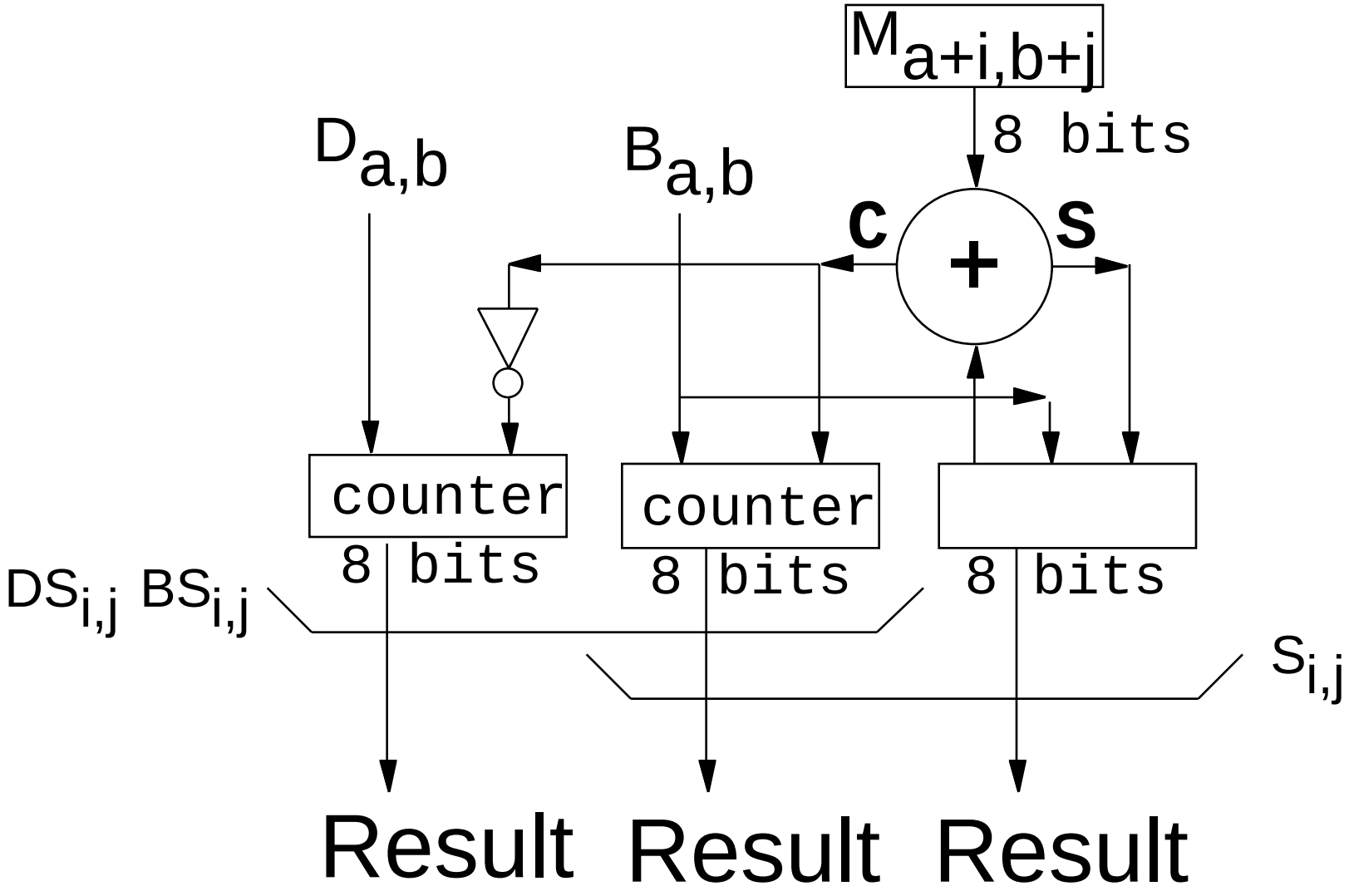


Three Computing Stages



Lots of reconfiguration opportunities!!

One Computing Configuration



Two Computing Phases

Phase-1 computes the 16bit $S_{i,j}$.

Phase-2 computes $BS_{i,j}$ and $DS_{i,j}$, each of 8bits.

The 16 bits results of each phase are returned over the fast pipeline.

The pipeline could convert the $S_{i,j}$ into $\Theta_{i,j} = (S_{i,j}/B_{count})$-Bias .

Additional Optimizations

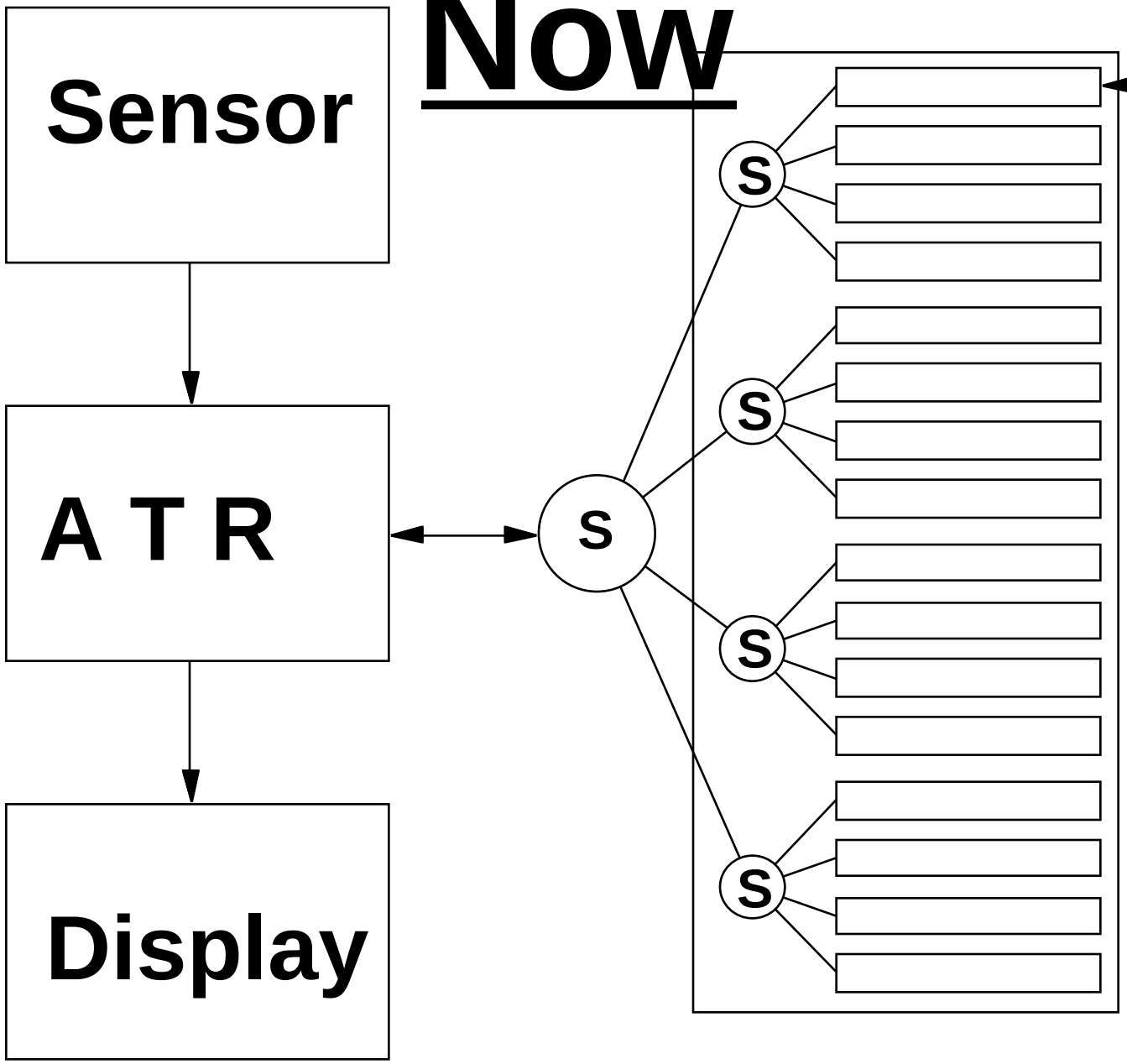
Track actual size of each template

Transpose tall templates into wide ones

Identify 0-rows of templates/ images

Early-outs (on Θ values)

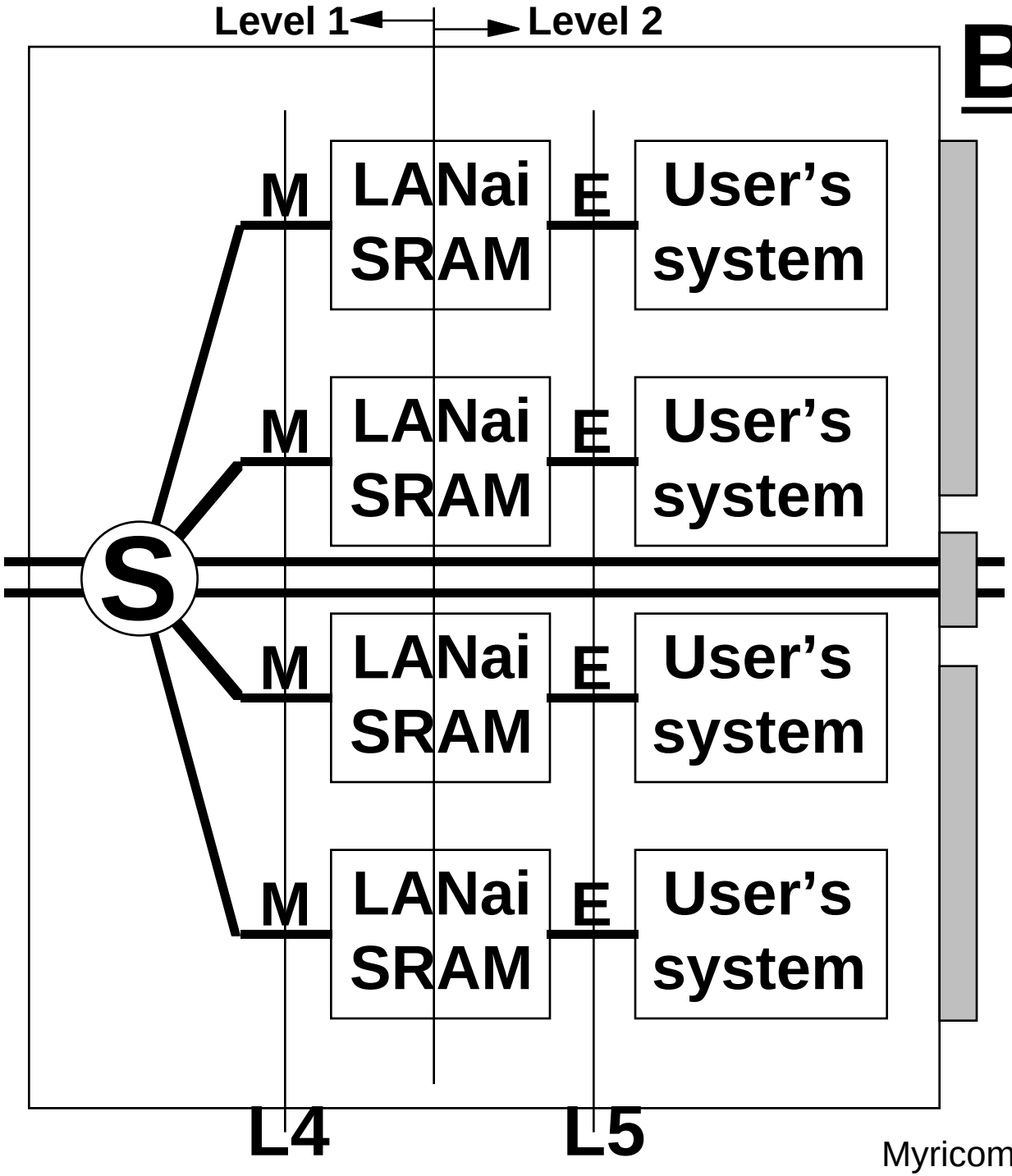
Now



Front Panel:

20 cards/subrack
16 Pcard/subrack
1 P/Pcard
700 (Match/s)/P
11,200 Match/s

Baseboards

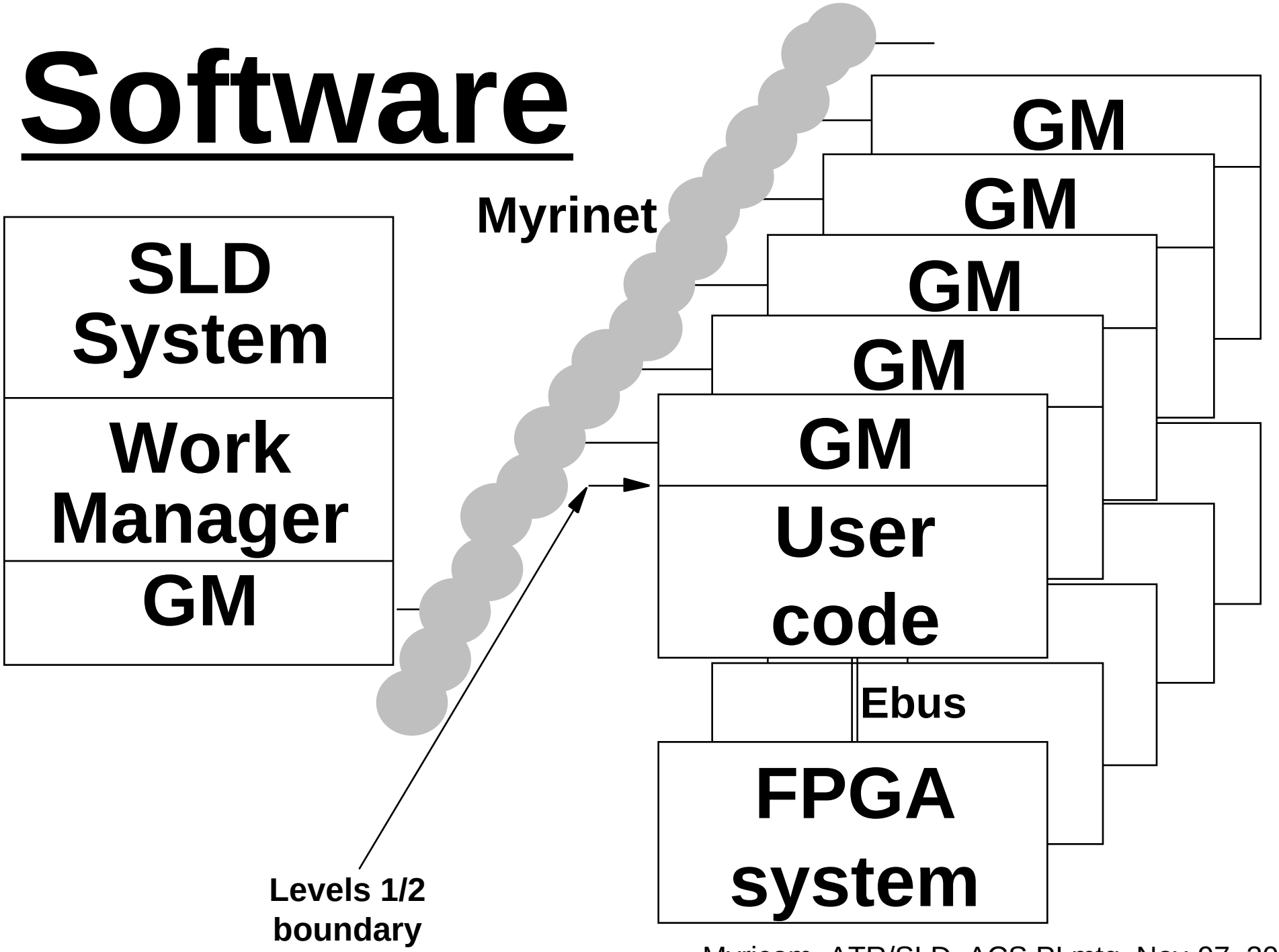


6U-VME card
4 ORCA nodes
2 Myri-links, front
2 Myri-links, back

Each L4-node:
1,300-1,600 matches/sec

~6,000match/card

Software


SLD System
Work Manager
GM
Myrinet
GM
GM
GM
GM
GM
User code
Ebus
FPGA system
Levels 1/2 boundary

Performance

Without optimizations:

32*32*21*2 = 43,008 op/match, overhead ~5K

40M(op/s) / 48K(op/match) = 833 (match/s)

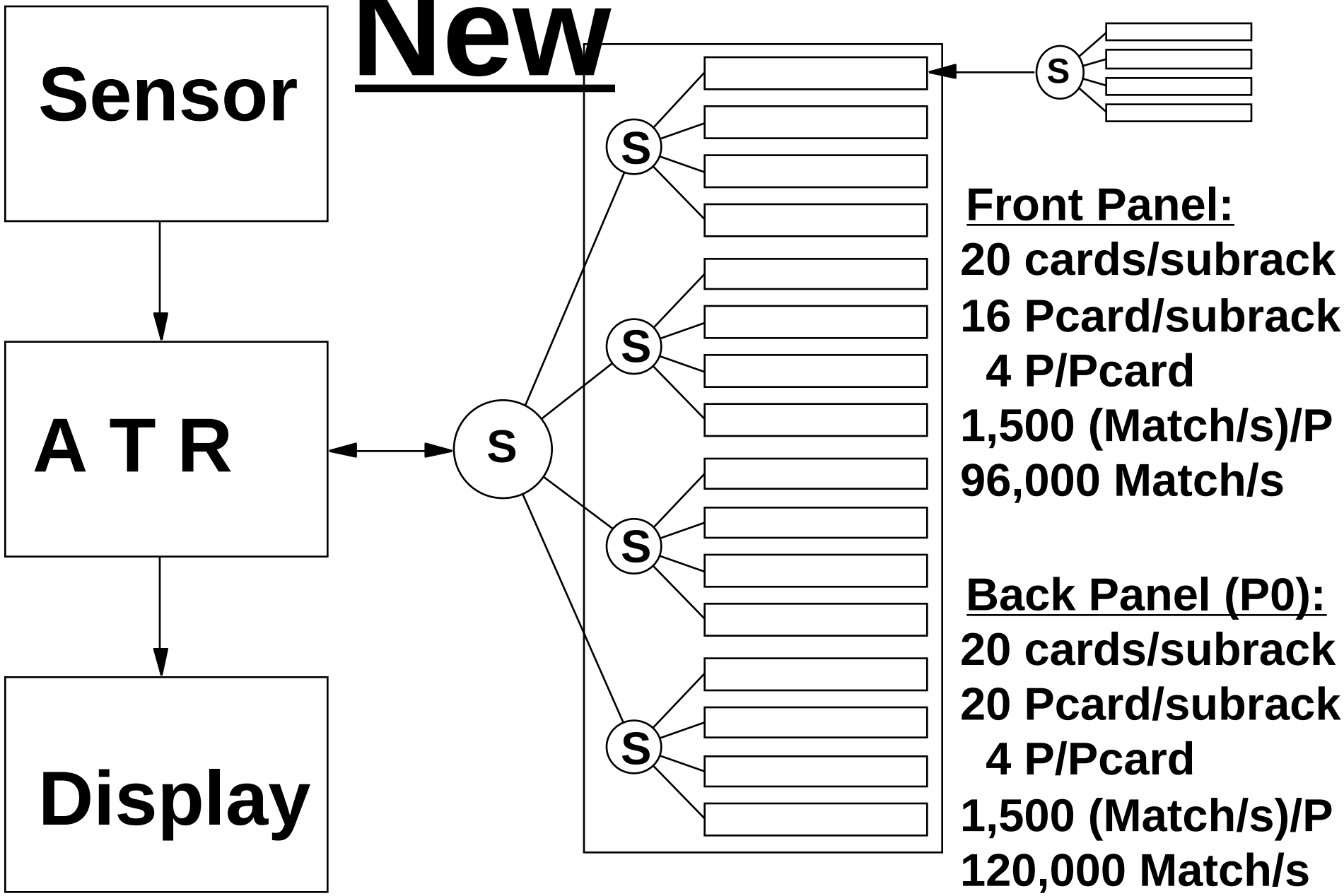
So simulated, and so measured.

With optimizations (but division by LANai):

1,300-1,600 (match/s) measured.

With division by FPGA: expecting 2,000 match/s

New



Front Panel:

20 cards/subrack

16 Pcard/subrack

4 P/Pcard

1,500 (Match/s)/P

96,000 Match/s

Back Panel (P0):

20 cards/subrack

20 Pcard/subrack

4 P/Pcard

1,500 (Match/s)/P

120,000 Match/s

Conclusion

To achieve high performance, use:

- Proper System Architecture
- Proper Decomposition HW+SW
- Dense Packaging
- Short inter-Reg path for fast clock
- High Parallelism
- Adapt to Dynamic Variances
- Every Cycle Counts
- Minimize Reconfiguration