# *Responsive Processor for Parallel/Distributed Real-Time Control*

Nobuyuki Yamasaki

Keio University / Electrotechnical Laboratory

E-mail: yamasaki@{ics.keio.ac.jp, etl.go.jp}

http://www.ny.ics.keio.ac.jp

# *Abstract*

*Responsive Processor* for Parallel/Distributed Real-Time Control integrates:

- Processing Core (SPARC)
- ***Responsive Link***(Real-Time Communication)
- Computer I/O (SDRAM I/F, DMAC, PCI, USB, PIO, SIO, Timers, Counters, …)
- Control I/O (ADC, DAC, PWM Generators, Pulse Counters, … )

System-on-a-chip

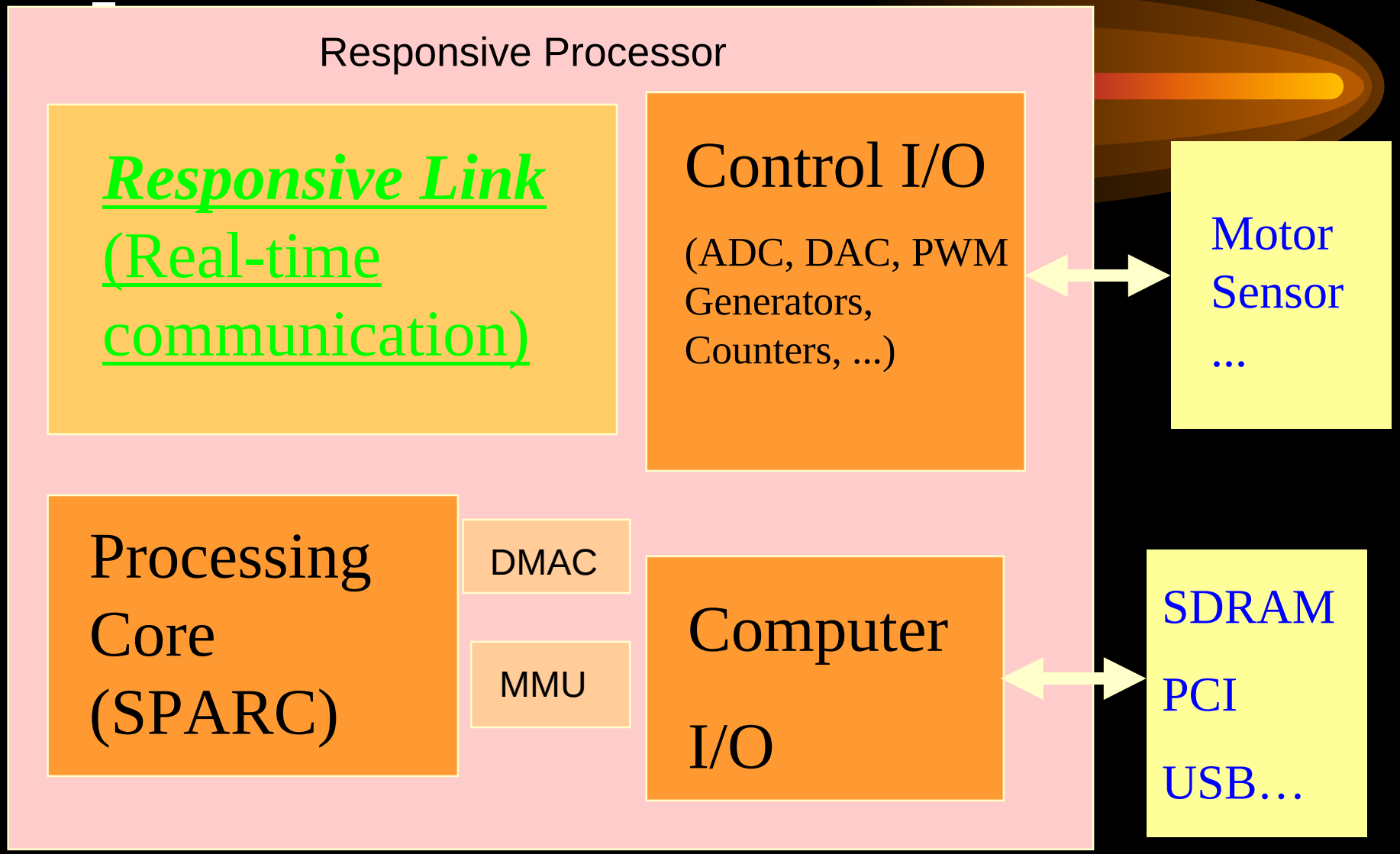
for Parallel/Distributed Real-Time Control

Combination with some processors

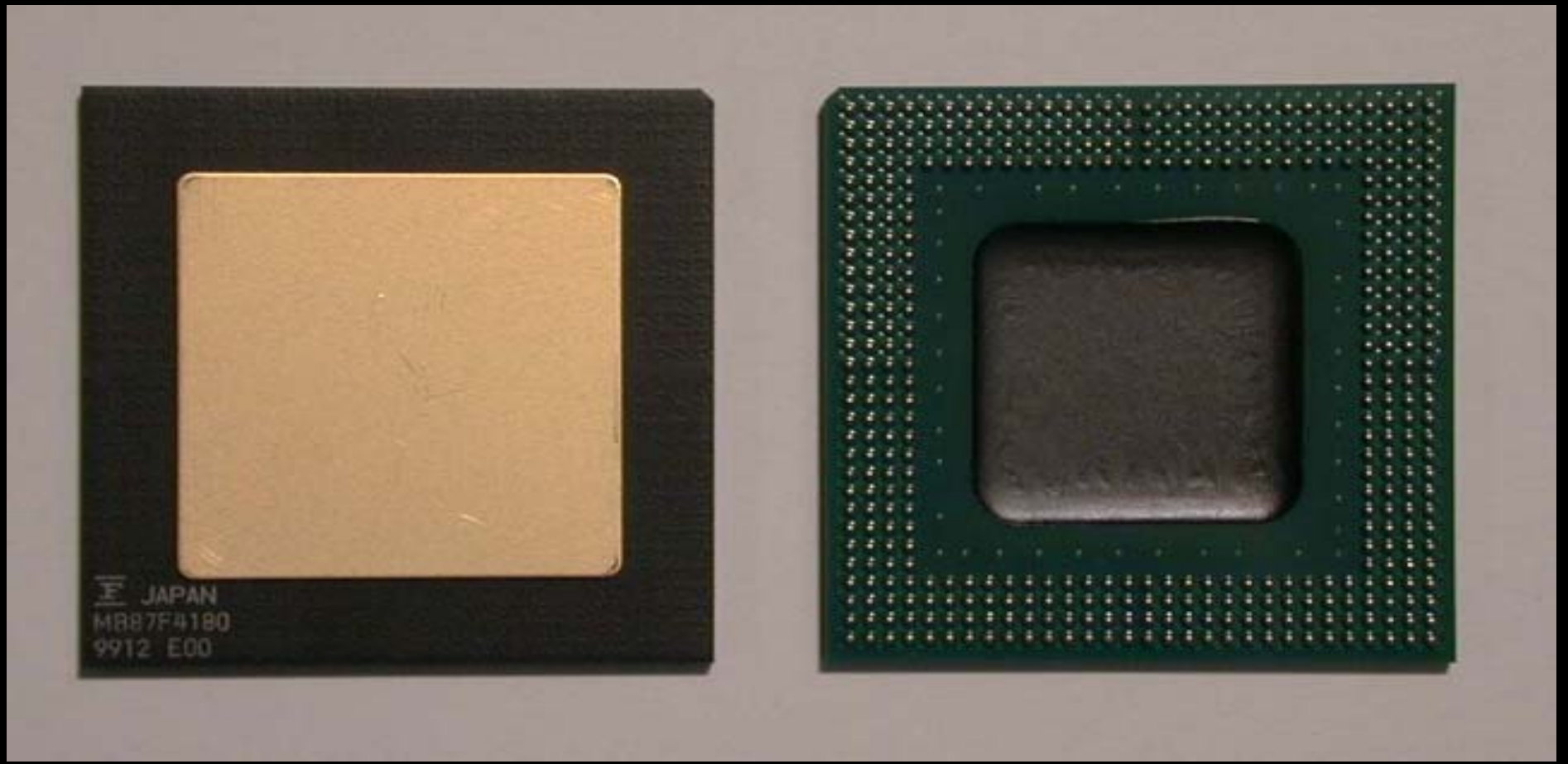
↓

Realization of various kinds of control systems

# *Components of Responsive Processor*



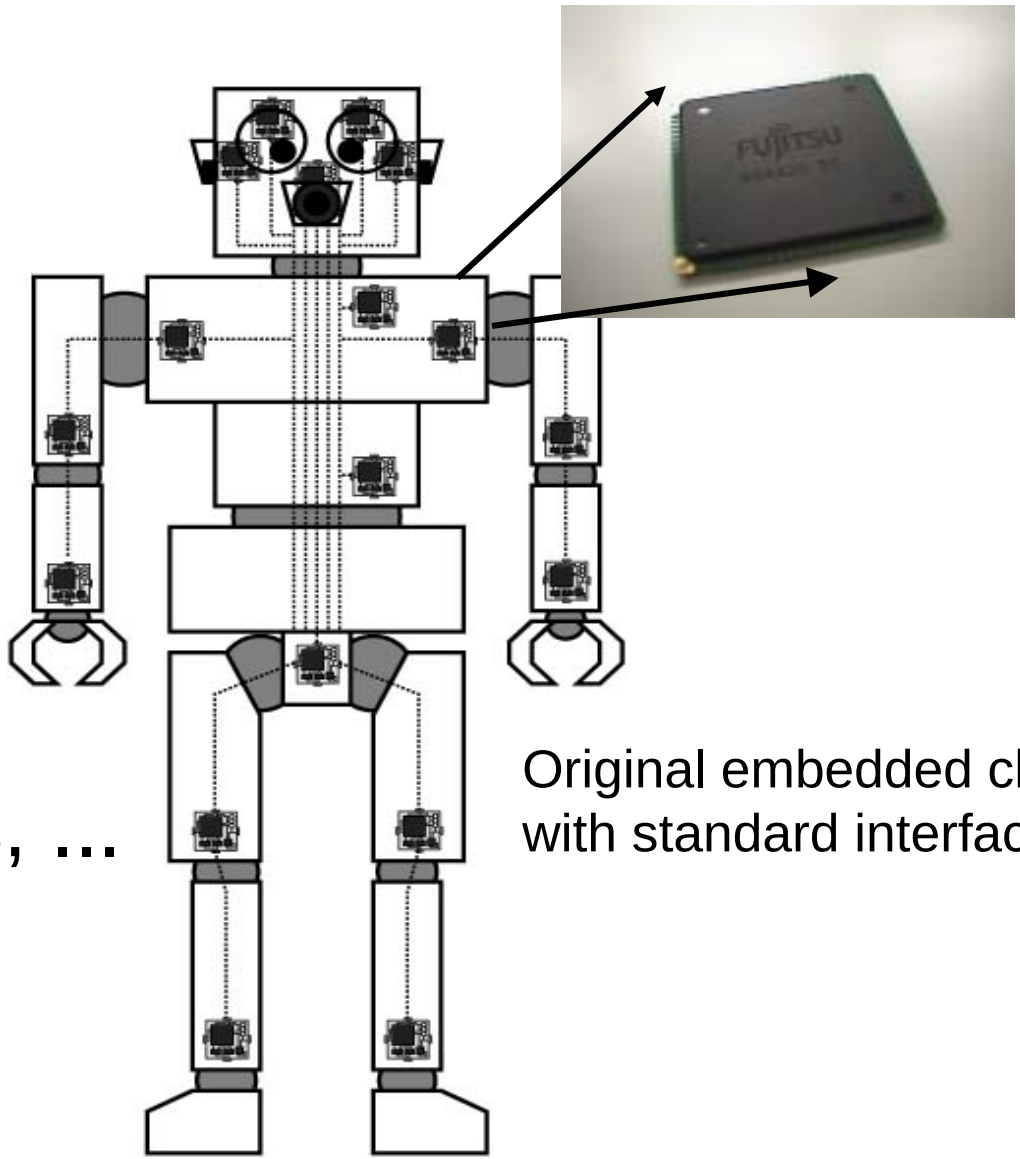
# *Responsive Processor*



# *Applications*

- Robots
- Office Automation
- Factory Automation
- Home Automation
- Intelligent Buildings
- Amusement Systems, ...

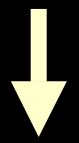
Functionally distributed control




Original embedded chip with standard interface

# *Event Processing for Embedded Control*

- Embedded control systems must respond to external inputs (sensors, user inputs, etc.) and operate according to the inputs.
- External inputs(interrupts) may occur frequently and continuously.



- Short response time
- Short processing time
- Control of devices(motors, sensors, etc.) → Time limitation.

# *Responsive System*

Responsive = Reactive + Real-time

- Reactive: A characteristic in which a reflexive action occurs according to an external input
- Real-Time: A characteristic in which the exactness of the system depends on not only the result but also the time it took to achieve the result

# *Design Policy of Responsive System*

## Basic Strategy by Hardware

- Shorter response time to external events
- Shorter processing time for the events

## Basic Tactics by Hardware

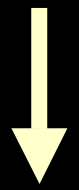
- Immediate transmission of events
- Parallel processing with respect to different kinds of events
- Concurrent processing in regard to the same kind of events

# *How to Realize a Responsive System*

- Generation of an event from a sensor
  - ➔Immediate transmission to the proper module to process the event
- Generation of several events simultaneously or continuously
  - Different kinds of several events
    - ➔Parallel processing by function-specific modules
    - ➔Reduction of both processing and response time
  - The same kind of several events
    - ➔Concurrent processing
    - ➔Reduction of response time

# *Function-specific Parallel Computer*

- Each module controls different I/O functions
- Coarse grain function-specific tasks
- Each module can be treated as an agent
- MIMD



Parallel computer architecture for embedded control systems

# *Current Real-Time Communication*

- IEEE-1394(FireWire, iLink, DV)
- USB
  - Soft Real-Time Communication : Isochronous transfer
  - Error correction is not supported in case of the isochronous mode.
  - Central Control : Low robustness
  - Maximum connection number is limited: ( IEEE-1394 : 63, USB : 127 )
  - Topology is fixed. (Tree structure)

# *Trade-off on Real-Time Communication*

- Soft real-time : Guarantee of band width
  - ➔Maximization of Throughput
- Hard real-time : Guarantee of latency
  - ➔Minimization of Latency

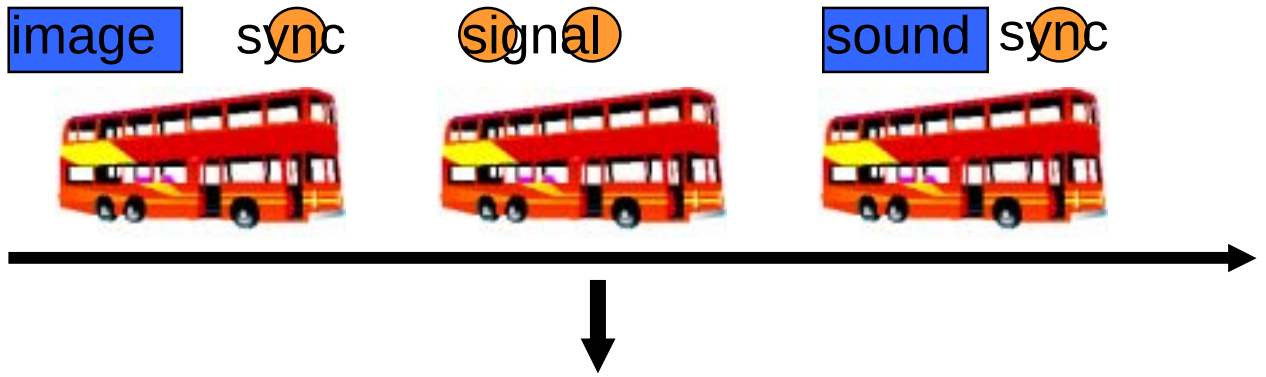
| ***PACKET SIZE*** | LARGE | SMALL |
|---|---|---|
| ***Throughput*** | High | Low |
| ***Latency*** | Long | Short |

# *Responsive Link for Real-Time Communication*

- Split transmission of data and events
- Fixed packet size：64B data, 16B event
- Full-duplex and differential I/F
  - Hardware routing
    - Independent routing of data and events
    - Cut-through switch with overtaking function (The packet with higher priority can overtake other packets at each node.)
    - Priority replacement (Packet priority can be replaced with new priority at each node.)
    - When the network address is same but priority is different, the different route can be reset to realize exclusive lines or detours.
  - Automatic error correction
- Flexible link speed (12.5 to 100M bps)
- Point-to-point link configurable for any topology



# *Split Lines for Event and Data*




Shared traffic

indefinite latency and throughput


Event link

Low Latency


Data link

High Throughput

## *Data Link*

Soft real-time communication for bulky data

- Multimedia data transmission (images, voice, etc.)
- Relatively large fixed packet size (64B)
- Total throughput is important

## *Event Link*

Hard real-time communication for control

- Inter-processor interrupt and synchronization
- Relatively small fixed packet size (16B)
- Low latency for relatively few packets is important

# *Packet Format*

**Data PacketFormat (64B)**

| Source Addr. | | Destination Addr. | |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | Payload | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | Control & Status | | |

**Event Packet Format (16B)**

| Source Addr. | Destination Addr. |
|---|---|
| Interrupt level | |
| Interrupt vector | |
| Control & Status | |

1 byte

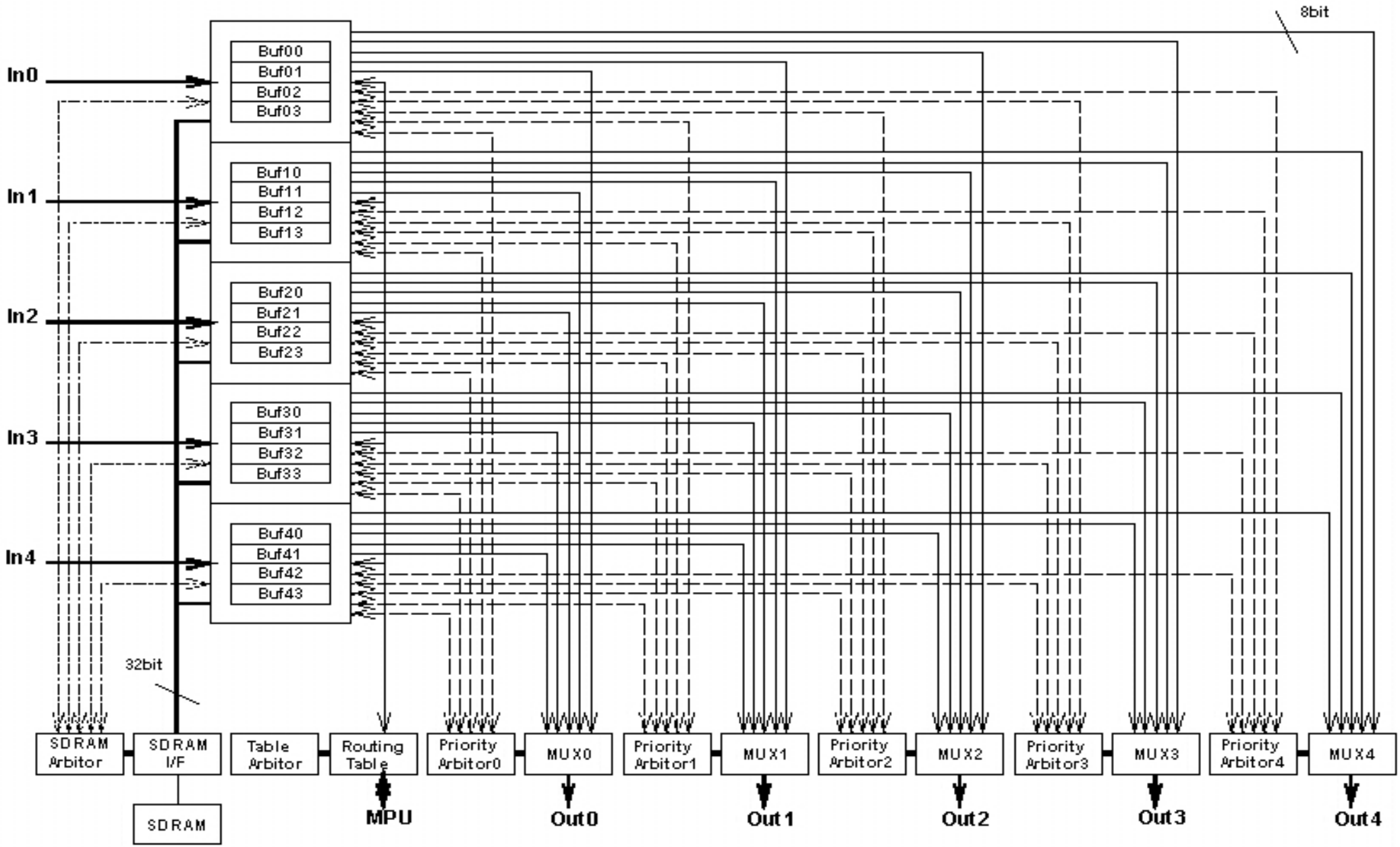
**Control& Status Format (32bits)**

| 1 bit | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | Full | | | Data Length | | |
| | 1 | Dirty0 | Dirty1 | Dirty2 | Dirty3 | Dirty4 | Dirty5 | Dirty6 | Dirty7 |
| | 2 | Dirty8 | Dirty9 | Dirty10 | Dirty11 | Dirty12 | Dirty13 | Dirty14 | Dirty15 |
| | 3 | Start | End | Int. | Fatal | Correct | Serial Number (Cnt.) | | |

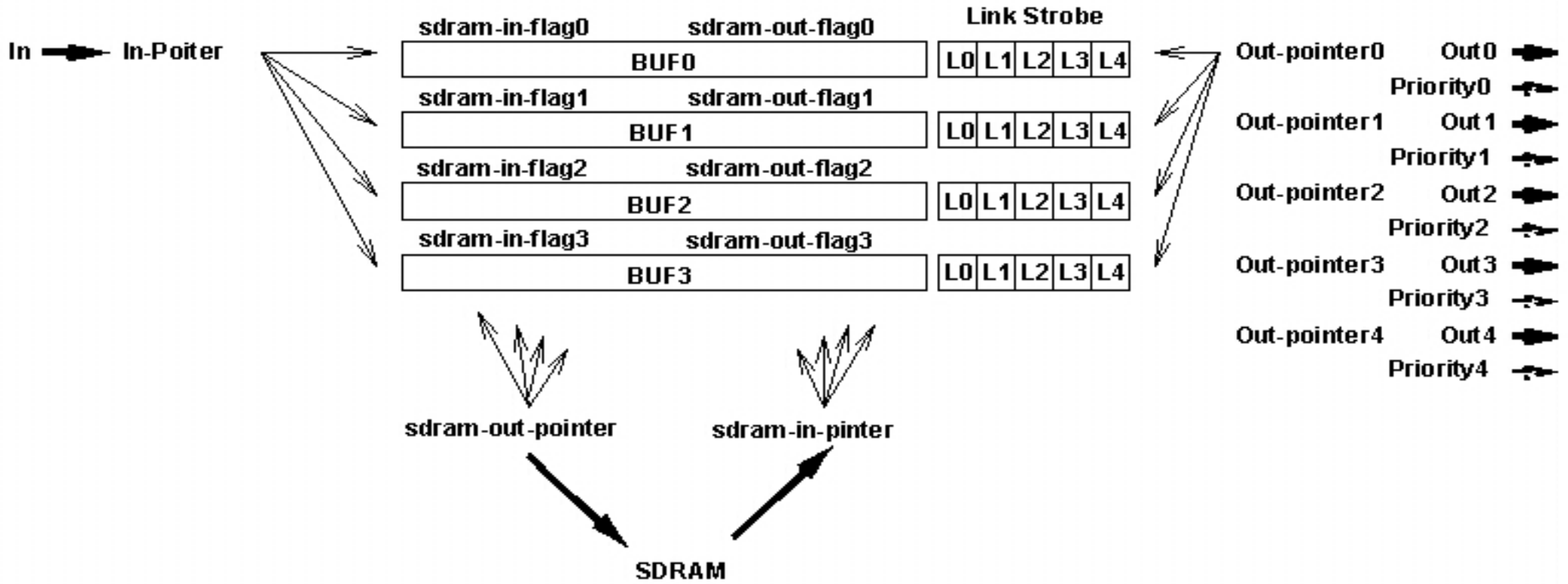
**Frame Format (14bits)**

| | | | Data bits | | | | | Redundancy bits | |
|---|---|---|---|---|---|---|---|---|---|

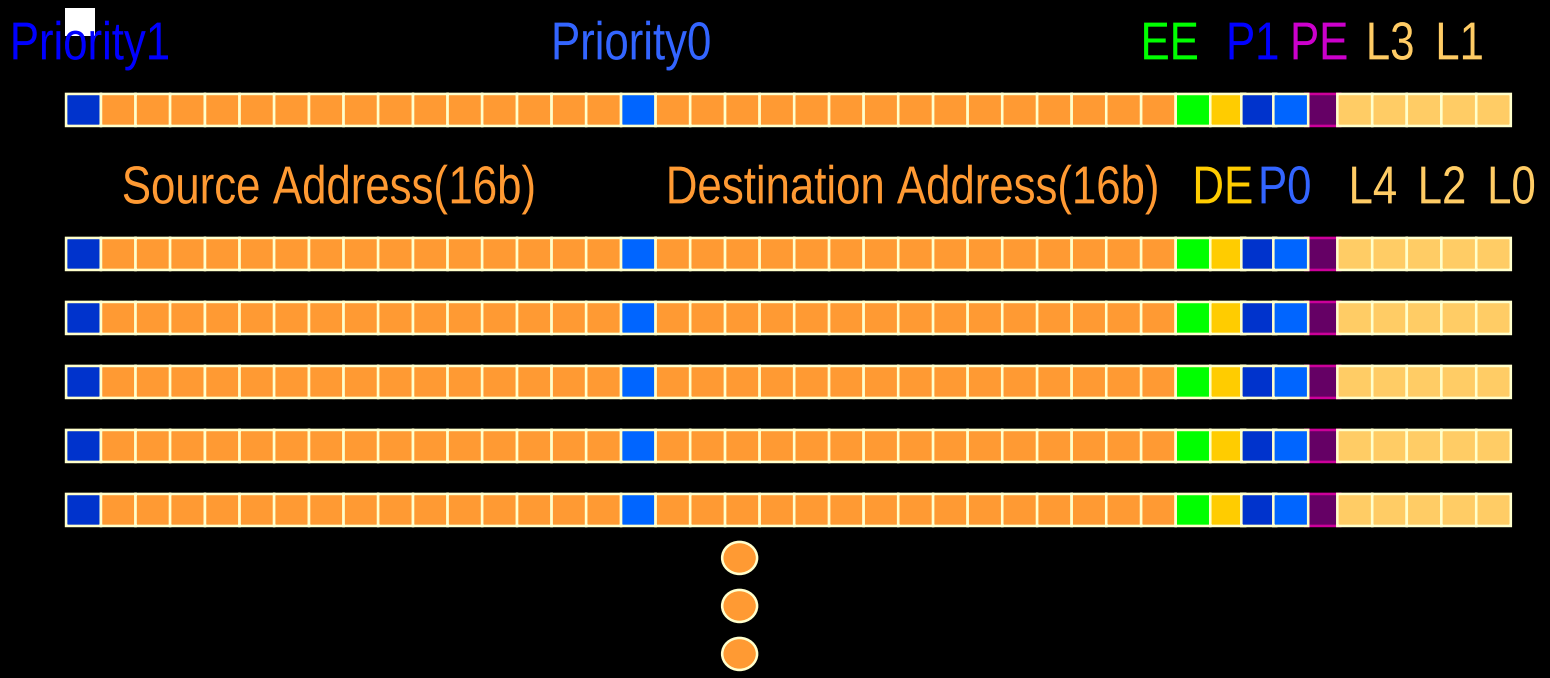
# *Cut-through Switch with Overtaking Function*



# *Control of Overtaking Buffers*

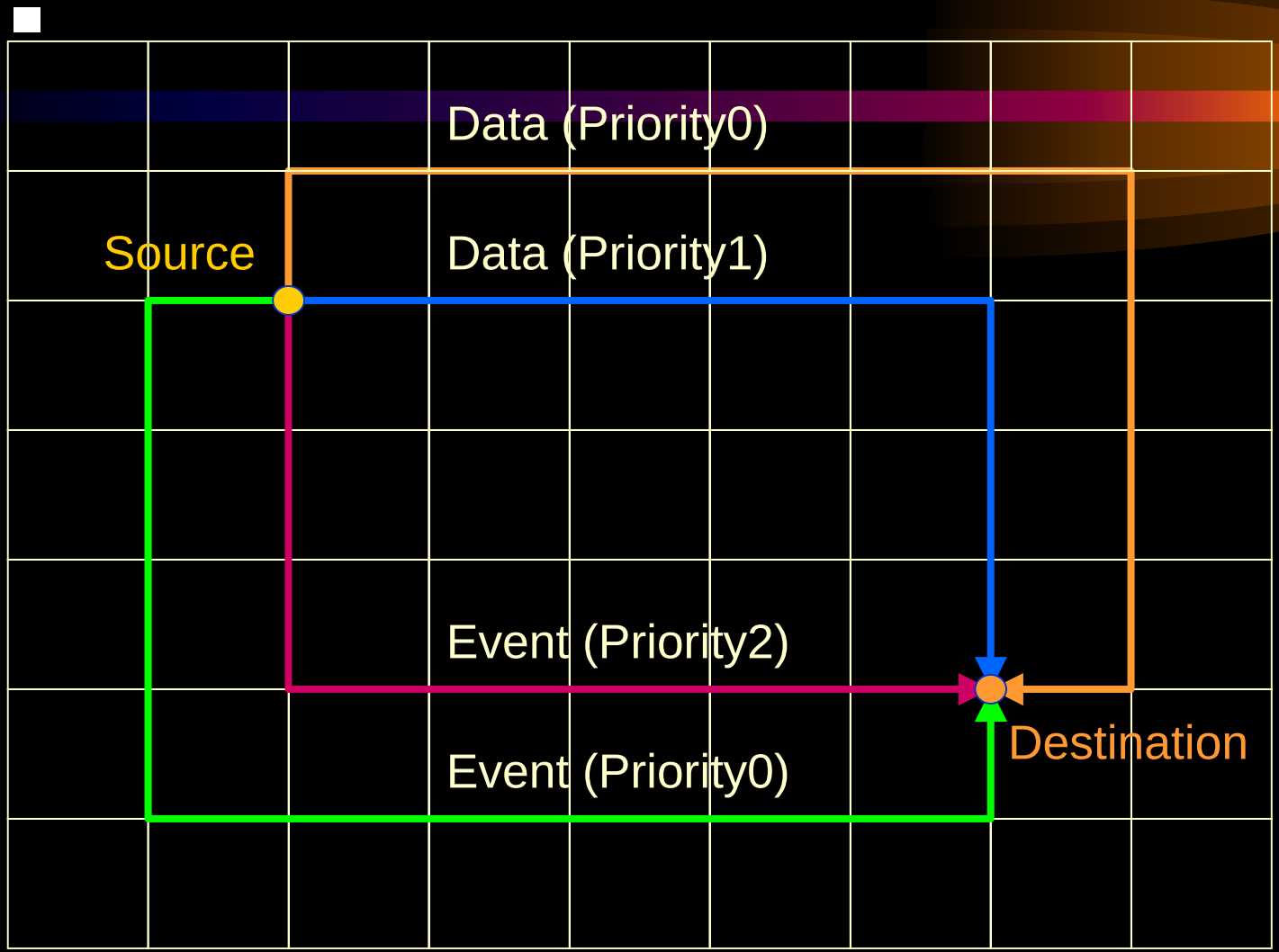


# *Routing Table*




- It is possible to set different routes in case of the same network address when the priorities are different. (Default route is priority 0.)
- It is possible to replace a packet priority with a new priority at each node.

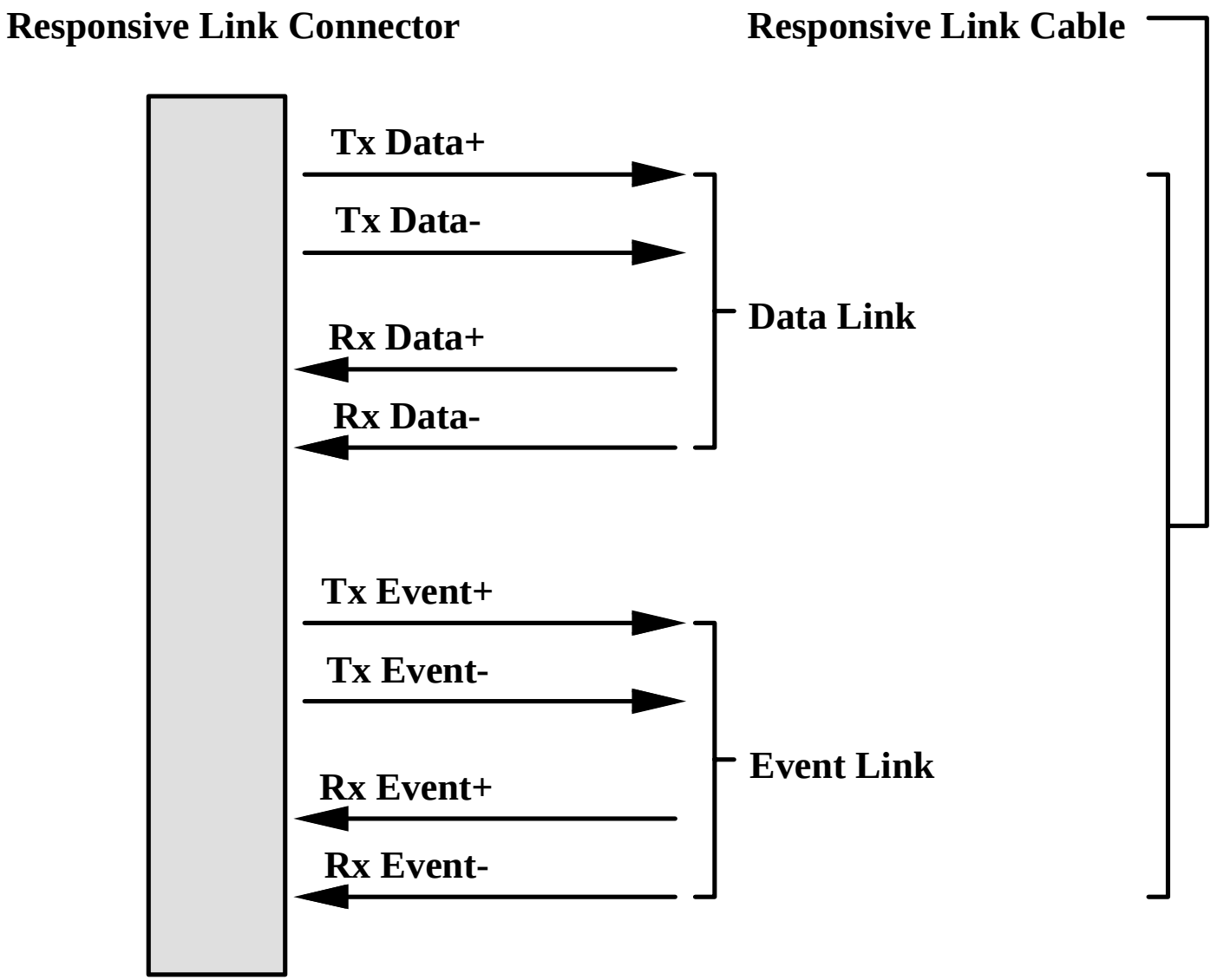
# *Routing according to Priority*



# *Low Level Communication*

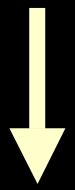
- Forward Error Correction(FEC)
  - Cyclic hamming code
  - 8bit data + 4bit redundant code
- Bit stuffing
- NRZI(Non Return to Zero Inverted)
- DPLL(Digital Phase-Lock-Loop)
- Synchronous frame (Setup Pattern)
- Flexible link speed (100Mbaud, 50Mbaud, 25Mbaud, 12.5Mbaud)

# *Responsive Link I/F*



# *Required Functions*

A questionnaire survey by using robotics and control mailing lists

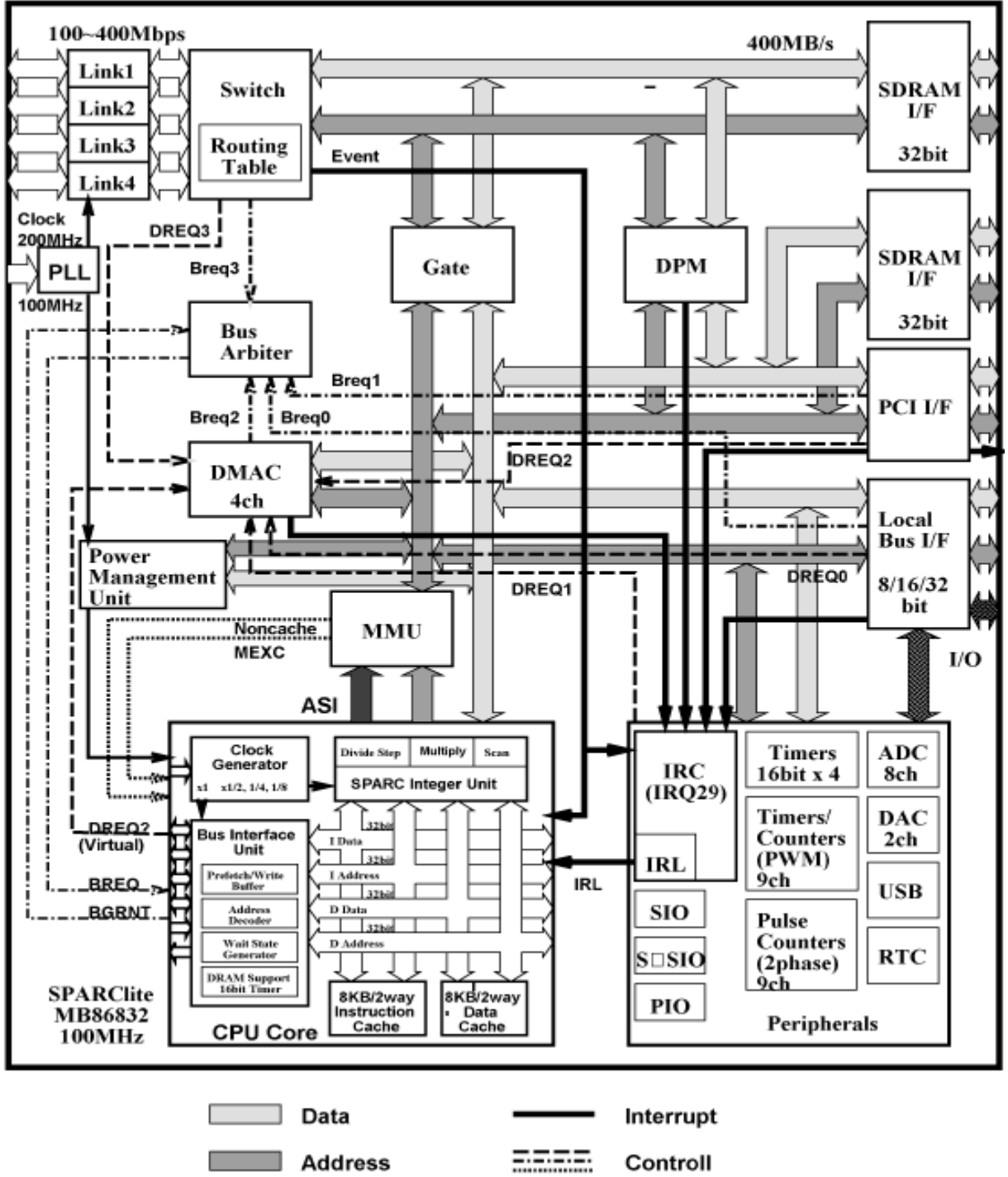


Required functions, developing environments, operating systems, etc.

# *Functions of Responsive Processor*

- Processing Core (SPARC 100MHz)
- Power Management Unit (100, 80, 60, 40, 20 [MHz], Sleep)
- MMU (64way)
- Responsive Links (4 links, 200, 100, 50, 25 [MHz])
- DMAC (4channels, Bus swapping, Bus sizing)
- SDRAM I/F (2channels, 100MHz)
- PCI I/F (Master/Target)
- USB I/F (Function, Hub)
- PWM Generators (50MHz, 9channels)
- Pulse Counters (24bit, 9channels)
- Timers/Counters (16bit, 4channels)
- Real-Time Clock
- A/D Converters (10bit, 8channels)
- D/A Converters (8bit, 2channels)
- Interrupt Controllers (43channels)
- SIO (RS-232C, 2channels)
- PIO (16bit), ...

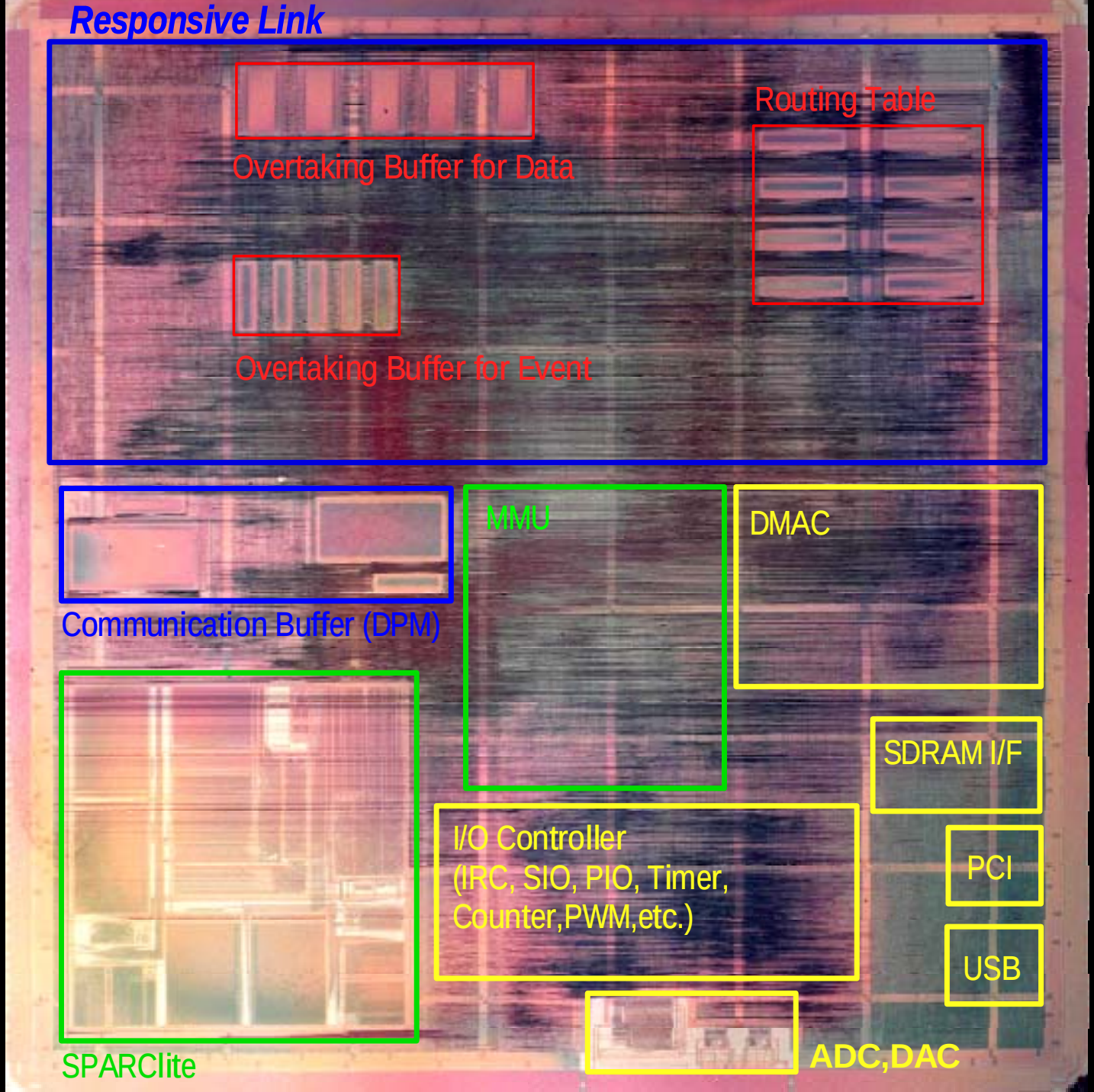
# *Diagram of Responsive Processor*



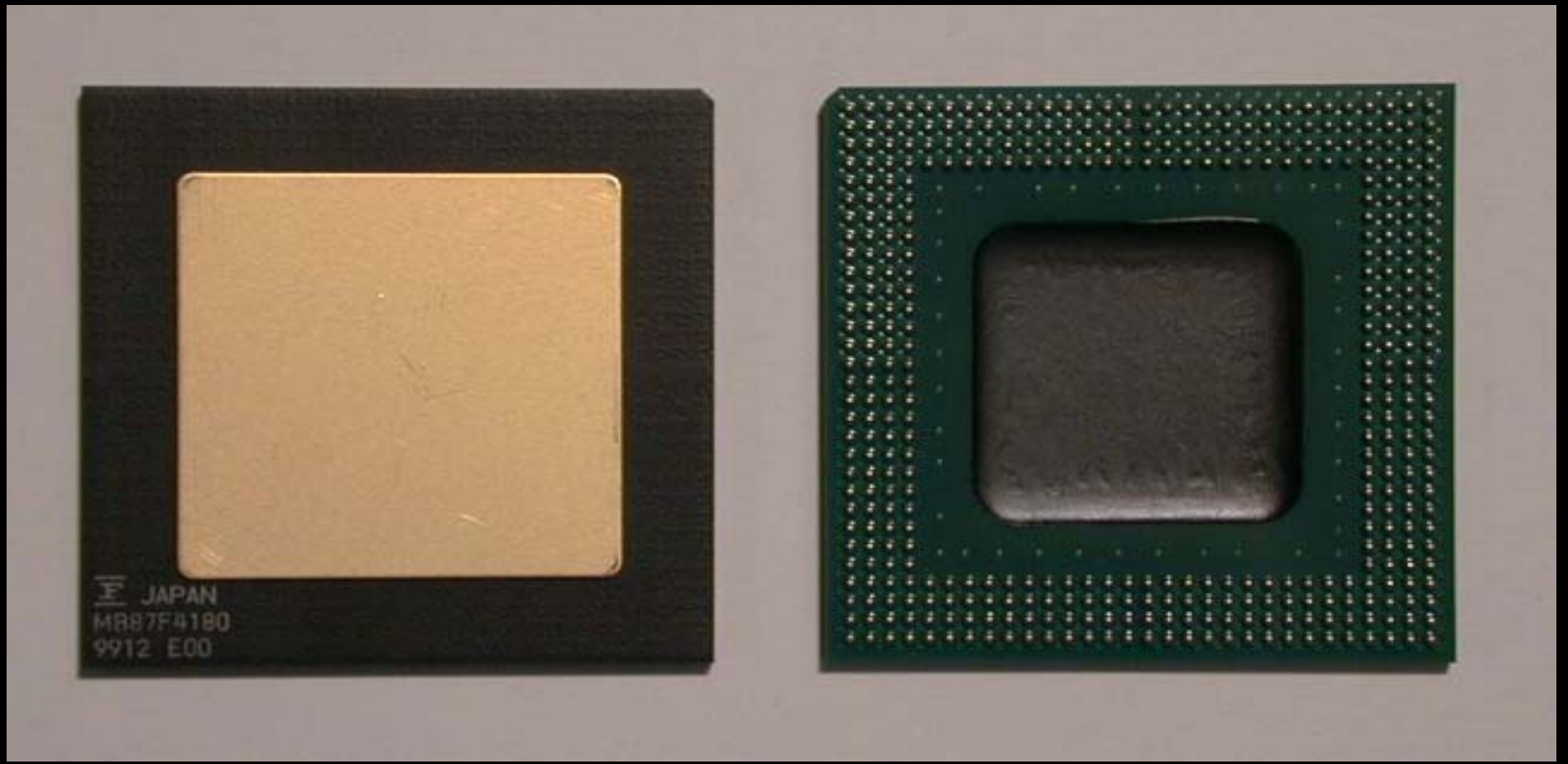
# *Hardware Design Rules*

- Process : 0.35 μm, CMOS, 4 layered metal
- Usable gates : 2,378 k gates
- Die size :14.5 mm x 14.5mm = 210mm2
- Package : 416pin BGA (40mm x 40mm)
- Voltage : 3.3V
- Max. power : 2W

# Layout


Responsive Link
Overtaking Buffer for Data
Routing Table
Overtaking Buffer for Event
Communication Buffer (DPM)
MMU
DMAC
SDRAM I/F
I/O Controller
(IRC, SIO, PIO, Timer,
Counter,PWM,etc.)
PCI
USB
SPARClite
ADC,DAC

# *Responsive Processor*



# *Performance (Speed v.s. Power)*

- Performance of MPU

| *Clock(MHz)* | *100* | *80* | *60* | *40* | *20* | *Sleep* |
|---|---|---|---|---|---|---|
| *Speed(MIPS)* | 121 | 97 | 73 | 48 | 24 | 0 |
| *Power(W)* | 1.0 | 0.8 | 0.6 | 0.4 | 0.2 | 0.01 |

Performance of Responsive Link

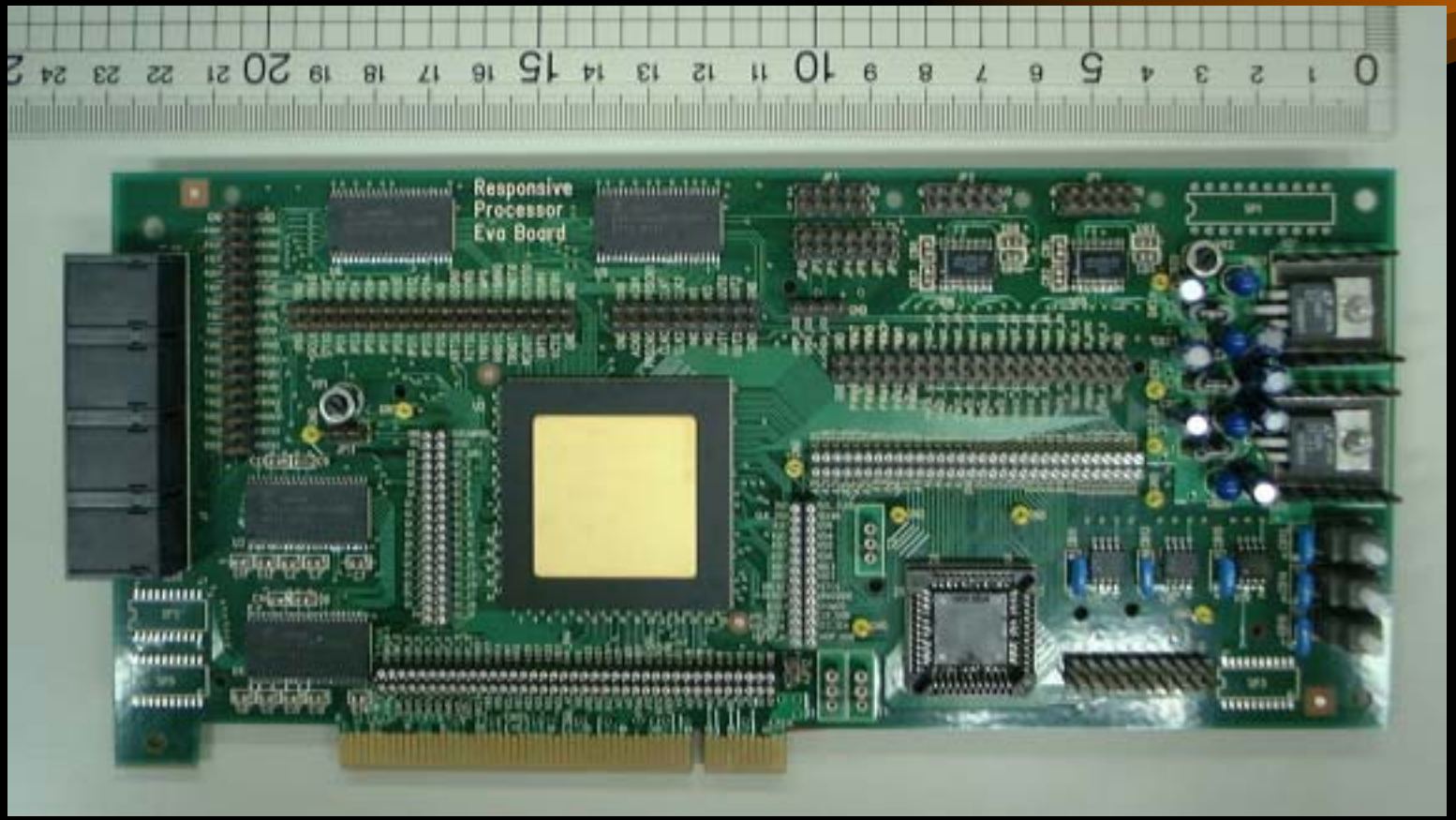
| *Clock (MHz)* | *200* | *100* | *50* | *25* |
|---|---|---|---|---|
| *Max. Speed (Mbaud)* | 100 | 50 | 25 | 12.5 |
| *Speed of Data (Mbps)* | 67 | 33 | 17 | 8 |
| *Latency of Event( $\mu$sec)* | 3.1 | 6.2 | 12.5 | 25 |
| *Power (W)* | 0.2 | 0.1 | 0.05 | 0.02 |

**Latency of Event (Worst) = 1 ( $\mu$sec) + 2 ( $\mu$sec/hop) x n (hop)**

# *ControlBoards*

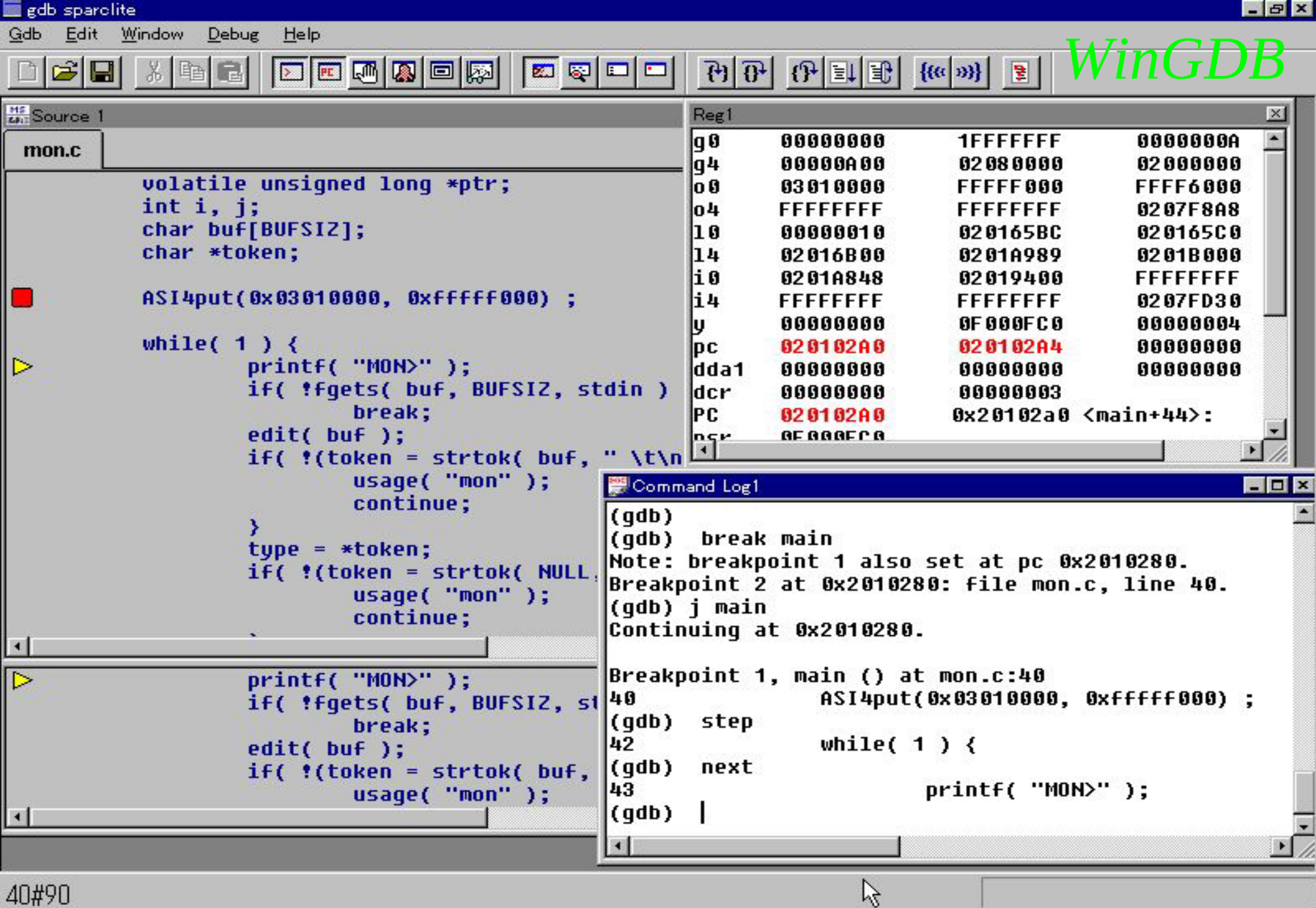
- PCI card (PCI half size)
- CardBus card (PCMCIA size)
- Embedded board (Credit card size)

# PCI Card



# *Developing Environment*

- Cross development on Windows PC via PCI, USB, or RS-232C
- GNU tools (gcc, as, ld, make, etc.)
- Source level debugging by GDB
- Host OS : Linux, FreeBSD, Solaris, Windows


gdb sparclite
Gdb Edit Window Debug Help
WinGDB
Source 1
mon.c
volatile unsigned long *ptr;
int i, j;
char buf[BUFSIZ];
char *token;
ASI4put(0x03010000, 0xffff000) ;
while( 1 ) {
printf( "MON>" );
if( !fgets( buf, BUFSIZ, stdin )
break;
edit( buf );
if( !(token = strtok( buf, " \t\n
usage( "mon" );
continue;
}
type = *token;
if( !(token = strtok( NULL,
usage( "mon" );
continue;
printf( "MON>" );
if( !fgets( buf, BUFSIZ, st
break;
edit( buf );
if( !(token = strtok( buf,
usage( "mon" );
Reg1
g0 00000000 1FFFFFFF 0000000A
g4 00000A00 02080000 02000000
o0 03010000 FFFFF000 FFFF6000
o4 FFFFFFFF FFFFFFFF 0207F8A8
l0 00000010 020165BC 020165C0
l4 02016B00 0201A989 0201B000
i0 0201A848 02019400 FFFFFFFF
i4 FFFFFFFF FFFFFFFF 0207FD30
y 00000000 0F000FC0 00000004
pc 020102A0 020102A4 00000000
dda1 00000000 00000000 00000000
dcr 00000000 00000003
PC 020102A0 0x20102a0 <main+44>:
Command Log1
(gdb)
(gdb) break main
Note: breakpoint 1 also set at pc 0x2010280.
Breakpoint 2 at 0x2010280: file mon.c, line 40.
(gdb) j main
Continuing at 0x2010280.
Breakpoint 1, main () at mon.c:40
40 ASI4put(0x03010000, 0xffff000) ;
(gdb) step
42 while( 1 ) {
(gdb) next
43 printf( "MON>" );
(gdb)
40#90

# *Operating Systems*

- Commercial
  - VxWorks
  - pSOSystem
  - µiTRON
  - OS-9
- Research
  - RT-Mach
  - ***µPULSER***
  - RT-Linux

# *Standardization*

## ISO/IEC JTC1 SC25 WG4 *Responsive Link* SG

***Responsive Link* SG**：**Matsushita Electric Industrial Co., Mitsubishi Electric Corporation, Fujitsu Limited, Hitachi, STARC, Electrotechnical Laboratory, Keio University, Kyusyu University**

**JTC1 SC25 WG4**：**RWCP, NTT, Oki Electric Industry, Yokogawa Electric Corporation, Mitsubishi Electric, Fujitsu Limited, Matsushita Communication Industrial Co., Toshiba Corporation, Hitachi Limited, SONY, NEC, Electrotechnical Laboratory, Nihon Unisys, Victor data systems, Japanese Standards Association, Sumitomo Electric industry**

# *Conclusions*

*Responsive Processor* for Parallel/Distributed Real-Time Control integrates:

- *Responsive Link*
- Processing Core (SPARC)
- Computer I/Os
- Control I/Os

Easy processor connection

Flexible configuration