New PrPMC evaluation

yamagata, T.Higuchi, and M.Nakao

COPPER requires PrPMC

- COPPER consists of
  - COPPER board
    - Belle2 will use COPPER-III
  - FINESSE
  - Trigger card “TTRX”
  - PrPMC to operate all other hardwares

EPC6315

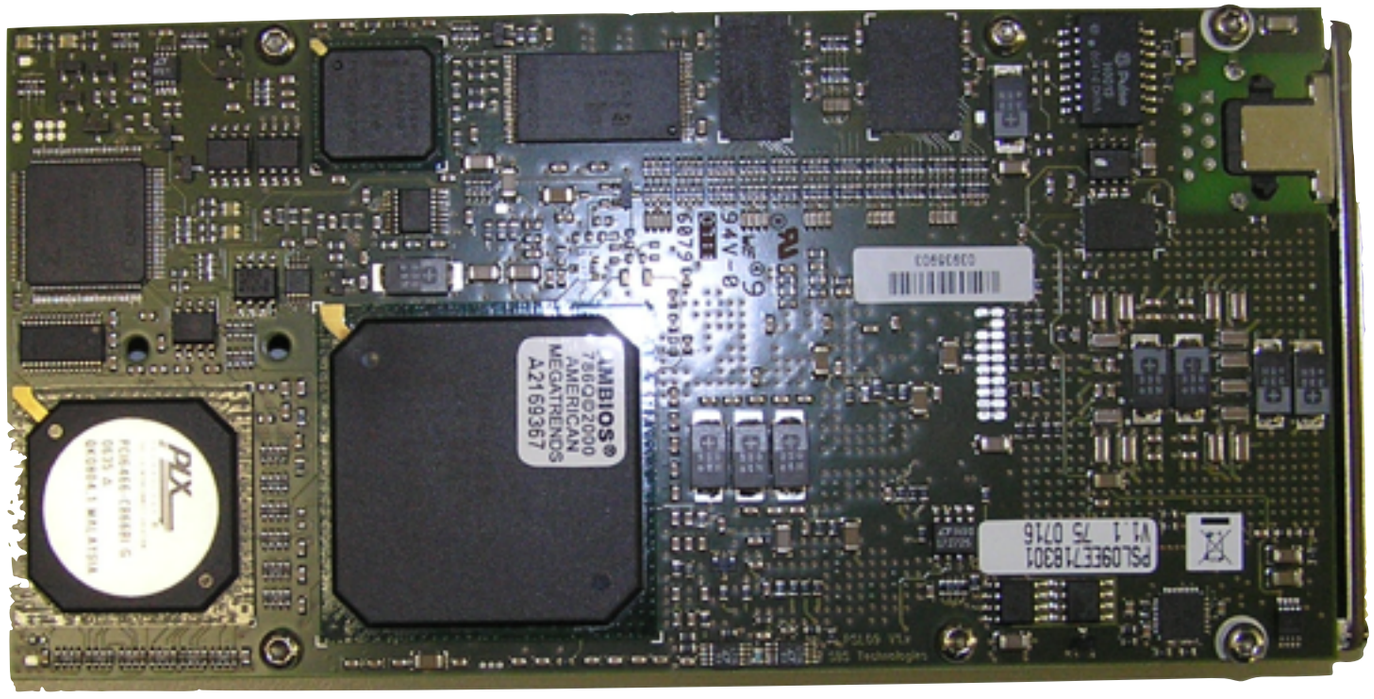
- Belle uses PrPMC EPC6315
  - PentiumIII 600MHz
  - 256M RAM
  - diskless boot
  - built-in e100

- But we don't want to use it any more

Why new one?

- EPC6315 is disappearing from the market.
- BIOS is not updated at all since 1st release.
- Vender aftercare is extremely poor.
- Difficult to keep contact with reseller in Japan.
- Low C/P (about 100k JPY)

PSL09



- PentiumM
- ESB6300
- e1000
- Throughput reaches the limit of 32bit PCI bus.
- Much expensive (about 200k JPY).

Then....

- We bid the development of new PrPMC.
  - Atom CPU
  - US15W chipset (Poulsbo)
  - Intel Gigabit Ether (not realtek!)
  - Must be driven by RH9 or RHEL3
  - co-works with COPPER-III
  - But must be cheap (about 100k JPY)

New features for our happiness

- VGA with COPPER
- KBD (USB or PS/2) with COPPER
- bootable from network
- bootable from USB CD drive
- bootable from flash (internal or CF)

Struggles of the company

- Power supply from COPPER
- BIOS
- Speed step (new)

Power supply

- COPPER supplies 5V + 3.3V
- PrPMC specification limits the current per pin.
- Atom + Poulsbo requires more power
  - DC-DC converter is necessary
  - but space of PrPMC is awfully small

Power-on sequence

- Atom + US15W has many power lines
- There is a correct sequence to turn-on of these lines.
- But it is not clearly described in anywhere.

If sequence is wrong,

- Electrical resistance becomes zero
- Too many current go through CPU + chipsets
- Burned them to broken
- Firstly they build three as the prototype
- Two of them burned

BIOS

- US15W can loads EFI or BIOS from SPIO interfaces
- The company doesn’t have SPIO aware BIOS
  - Mostly EFI is used for SPIO
- bridge chip for BIOS-ROM is required

SpeedStep

- SpeedStep is a technology by Intel to reduce the power consumption. It changed CPU clock dynamically.
- If it is turned on BIOS (default: ON), the system will freeze during CPU is fully consumed by user processes.
- Solution
  - BIOS default has changed (default: OFF)
  - One capacitor will be replaced to avoid the freezing with SpeedStep ON.

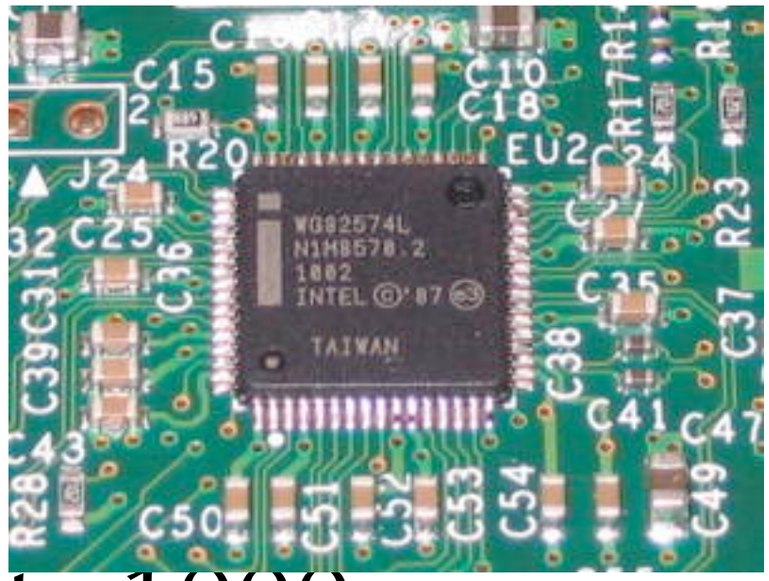
About Linux kernel

- We will use linux 2.4 for Belle II.
- Yes, we want to use linux 2.6.
  - Tick for task switch is 10ms in 2.4, 1ms in 2.6.
- but I don't have enough time to rewrite and validate the COPPER driver for 2.6.
  - It uses deprecated kernel functions
  - other drivers (TTRX + AMT3) seems to work well, confirmed
- This requires several dirty works for network boot and ethernet handling.

Ethernet

- We surveyed ethernet chip of Atom PC/ boards in the markets
- 1st: realtek, which consumes CPU.
- 2nd: 82574 family, slightly expensive.
- 3rd: USB ethernet, netbooks.
- We choose 82574 to save CPU consumption

82574



- e1000 doesn't handle it, but e1000e does.
- Also GRUB-0.9x doesn't handle it.
- e1000e can't be statically linked in linux kernel.
- Traditional network boot scheme requires statically linked ethernet driver.

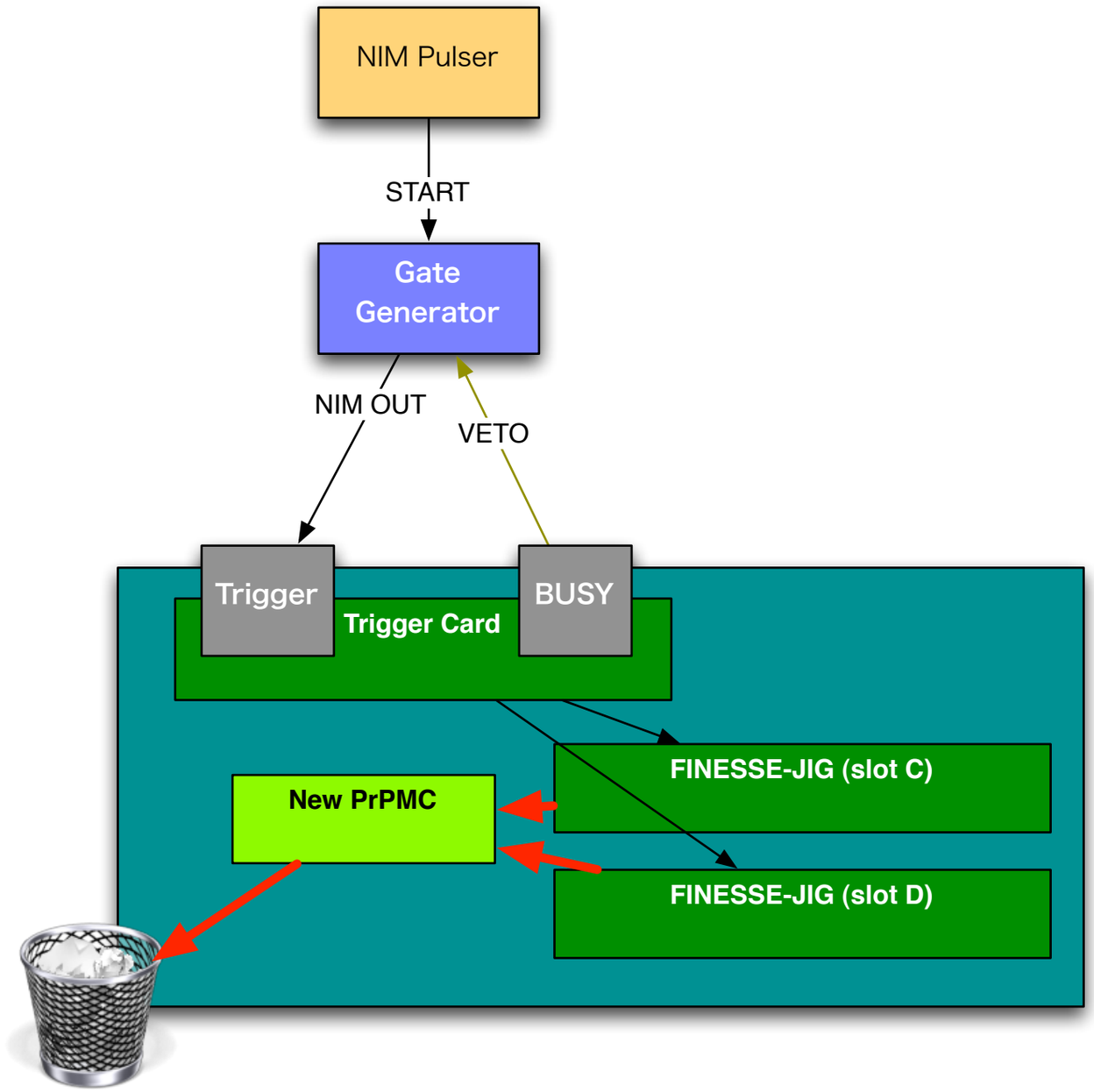
Patchy scheme

- e1000e driver is loaded via initrd
- linuxrc on initrd assign IP addr by
  - /sbin/ifconfig (yamagata)
  - /sbin/dhclient (Company, kernel differs from that for EPC6315)

Performance test

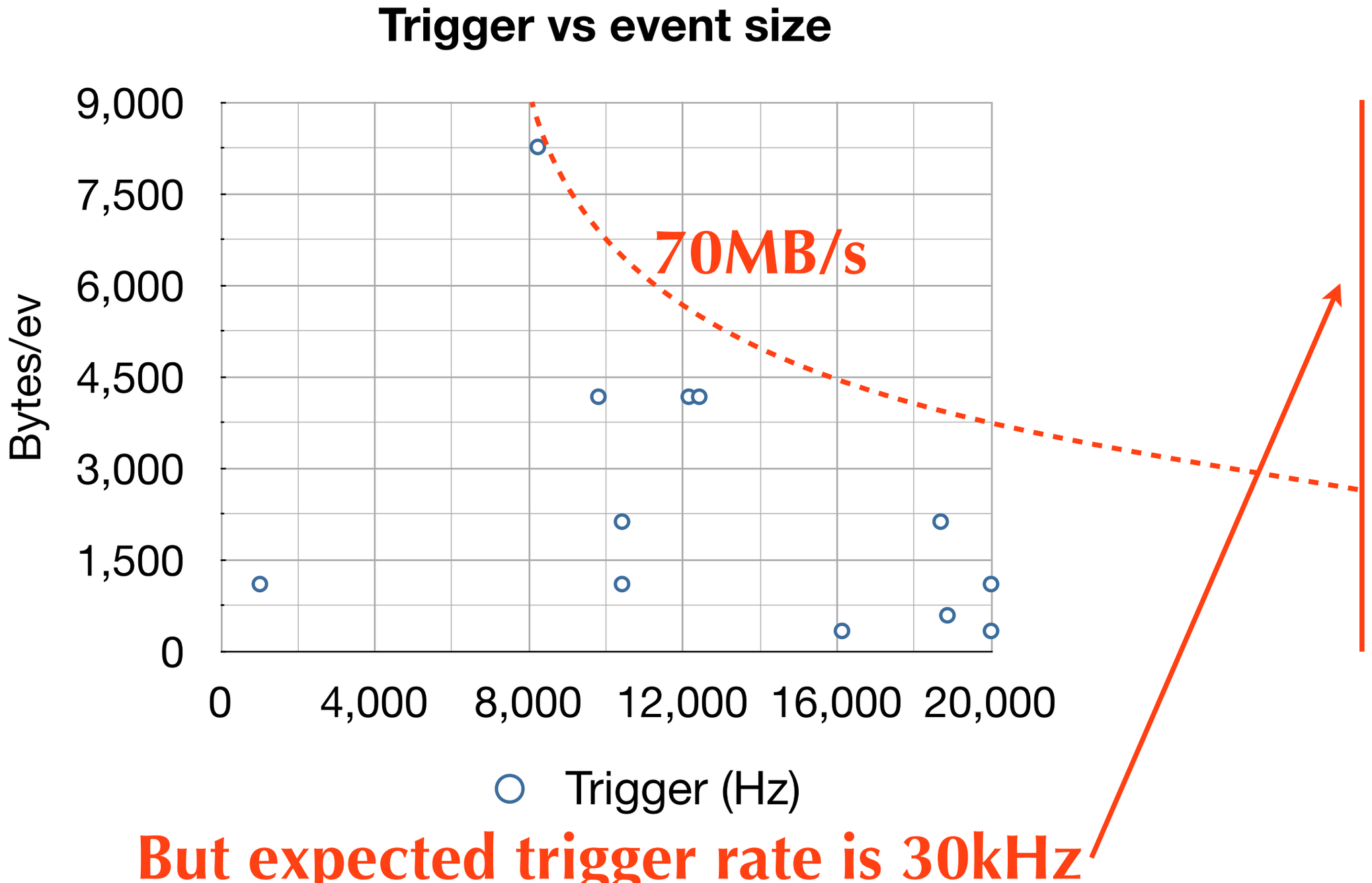
- FINESSE-JIG + NIM trigger
  - Trigger card is not TTRX, but another one for J-PARC experiments.
  - External NIM signal as trigger
  - BUSY signal is extracted as NIM signal
  - we can confirm the low level behavior of COPPER and FINESSEs.

Configuration



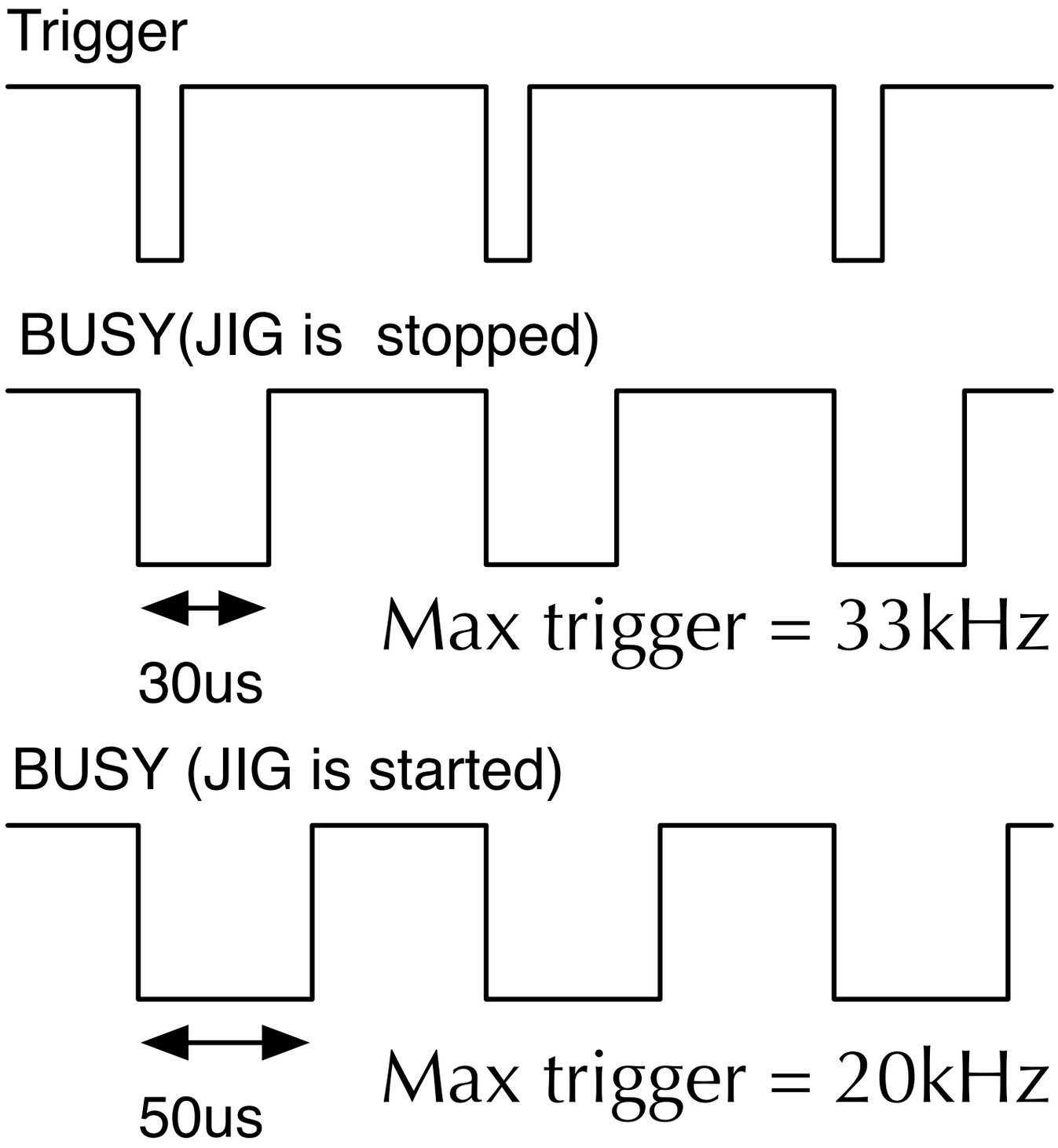
Data is **not** sent to network

Result (at last autumn)



Why the trigger rate is slow?

BUSY signal from Trigger card



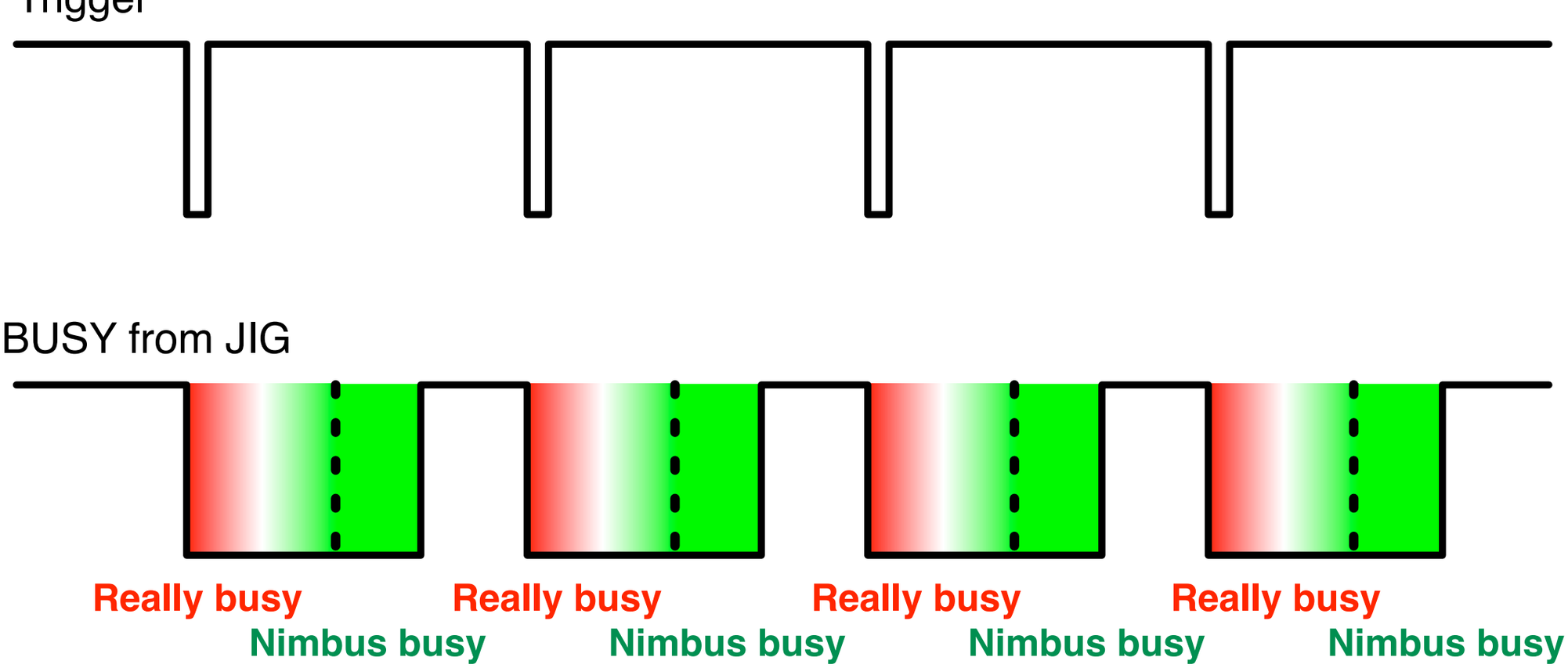
Maybe Trigger card problem, replace it to TTRX

Still BUSY is long

- With the correct BUSY handshake, maximum trigger rate is only 24kHz.
- Why JIG generates so long BUSY?
- Anyway, really JIG is busy during the BUSY signal is active?

Guess “Nimbus BUSY”

- JIG keeps BUSY active longer than actually it is busy.



Ignoring BUSY from FINESSE

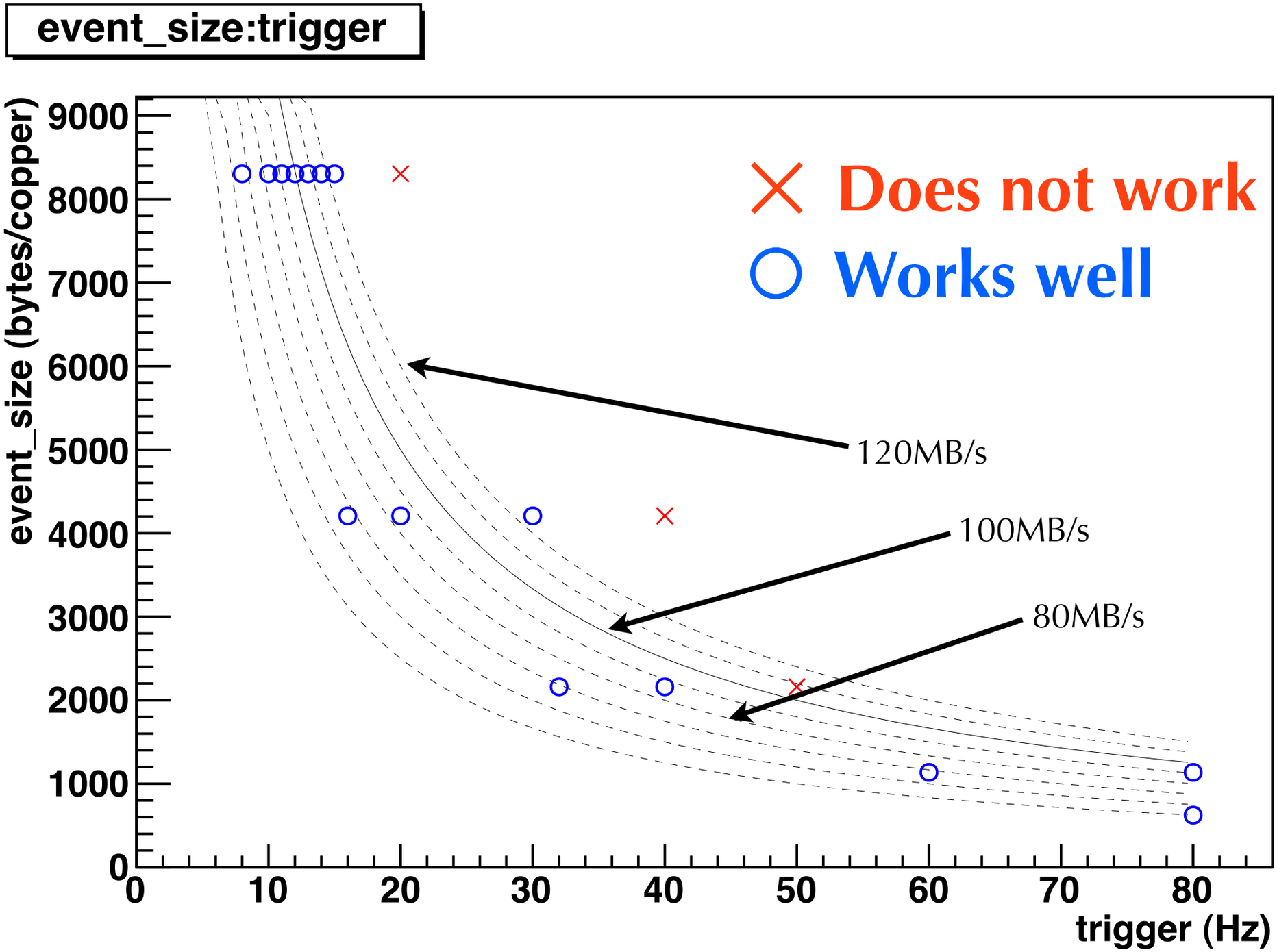
- With
  - ignoring from FINESSE JIGs
  - periodical trigger generated by TTRX
  - checking the consistency of
    - # of events from FINESSE-JIG
    - # of trigger from TTRX
  - checking checksum, header, footer.

- If # of events are consistent, FINESSE-JIG can accept next trigger during NIMBUS BUSY.

- Yes, the guess is correct, even though the length of NIMBUS is not clear.

- We can test throughput with the faster trigger.

Result


event_size:trigger
event_size (bytes/copper)
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
10
20
30
40
50
60
70
80
trigger (Hz)
Does not work
Works well
120MB/s
100MB/s
80MB/s

How about CPU power?

- New PrPMC has built-in GbE
- Can it sends rawdata from FINESSE without reduction to network?
- 1Gbps traffic requires 1GHz CPU processing power of Pentium III and Intel GbE NIC.
  - This heavily depends on Network controller.
  - Cheap NIC consumes more.
- Processing power of ATOM is smaller than that of PentiumIII

Brief test

- 8kB/ev data * 8kHz trigger => 64MB/s throughput
- Two process model
  - read COPPER and write it to stdout
  - send data from stdout to another host via network
  - concatinated by pipe
- CPU idle is 18%
  - It is about 62% when the first process discards data without writing stdout (not writing to /dev/null)
- 1Gbps will use (62 - 18) * 100 / 64 = 68.5 % of CPU
  - it is larger than 62%
  - 1Gbps reading COPPER + transmission to network may not be possible.
- Maybe it will decrease using large MTU

Summary

- New PrPMC seems to be better than EPC6315
  - Faster CPU power
  - Faster NIC
  - Higher throughput
  - Good response from vendor and reseller
  - Better C/P