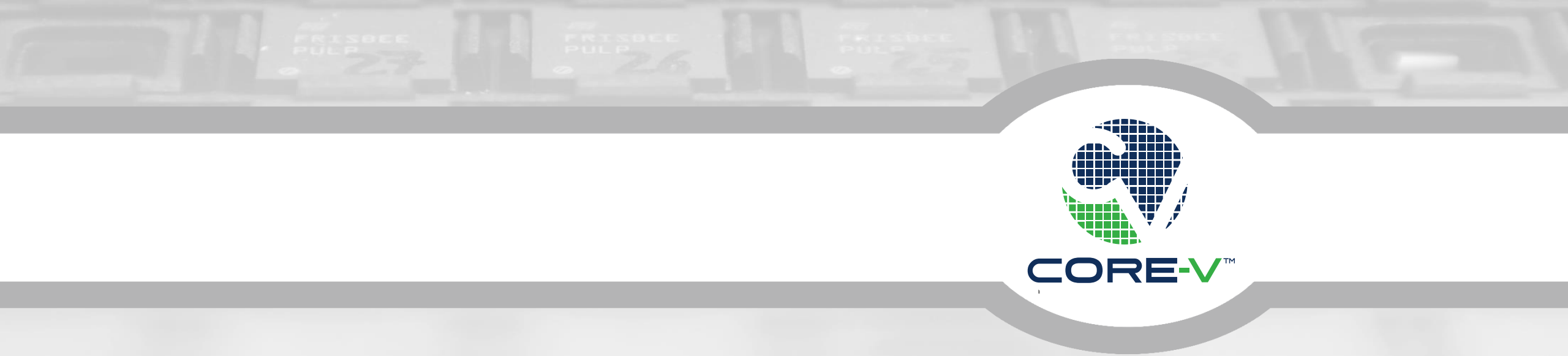


CORES TG – July 5 2021

Arjan Bink

Jérôme Quevremont

Davide Schiavone

Agenda

- SFU – TAIGA processor
- Dolphin – CV32E40P (v2)
- CVA6 status
- CV32E40X status

CV32E40P v2

Olivier Monfort, Pascal Gouedo, Yoann Pruvost
Dolphin Design
July 5, 2021

Final CV32E40P v2 features choice

- Re-Encode in Custom extension all PULP “basic” instructions and Verify them
  - Post-increment and register-register indexed load/store
  - Hardware Loops
  - General ALU extensions
  - Immediate Branch
  - Multiply and Accumulate

- Re-Encode in Custom extension all PULP instructions and Verify them
  - Bit Manipulation
  - 16- and 8-bit SIMD
  - **No** addition of 4- and 2-bit SIMD version

Final CV32E40P v2 features choice

- Re-Encode in Custom extension and Verify PULP instructions used for Cores synchronization
    - PULP Event load

- For Multicore cluster area reduction
    - Verify PULP Zfinx

- Floating Point Unit
    - Keep FPU instructions decoding inside the Core
    - Dispatch them using APU interface

⇨ With the listed features, E40P name can be kept as it complies with rules.

Status

- Started to setup core-v-verif environment w/ Mike Thompson support

- Started contacting Imperas
    - to get CV32E40P v1 Reference Model license to start running UVM environment
    - to add Custom PULP instructions into Reference Model

- Continue to talk with Embecosm to add Custom PULP instructions into GCC SW toolchain

DOLPHIN

DESIGN

CVA6 status

Jérôme Quévremont

CVA6

- Gates
  - PC (former PPL) passed
  - PL passed
  - PA (former PPA) to come: specification and detailed plan
- Project meetings
  - Every Friday 14:00 CEST
  - Alterning technical and progress meetings
  - https://calendar.google.com/calendar/u/0/embed?src=meetings@openhwgroup.org
- TWG : Cores : CVA6 Mattermost channel
  - Also spans on verification, SW topics…

CVA6: current activity

- Specification on going (link)
  - Recent progress:
    - CSR map
    - Performance counters optimizations: 6 generic counters, 22 possible sources
    - Work with CV32E40X to get the same coprocessor interface specification
  - Close to a complete version
- Design: investigating how to merge Sv32 and Sv39 MMU in a common file (SW parameters not sufficient).
- Verification: rearranged repo to clearly delineate the core (=scope of the specification) and the APU (processor subsystem)
- SW: on-going FreeRTOS and 32b Linux ports
  - 32b Linux demo on CV32A6 (BuildRoot 2021.5.rc1, Linux kernel 5.10.7)

Custom CSR mapping

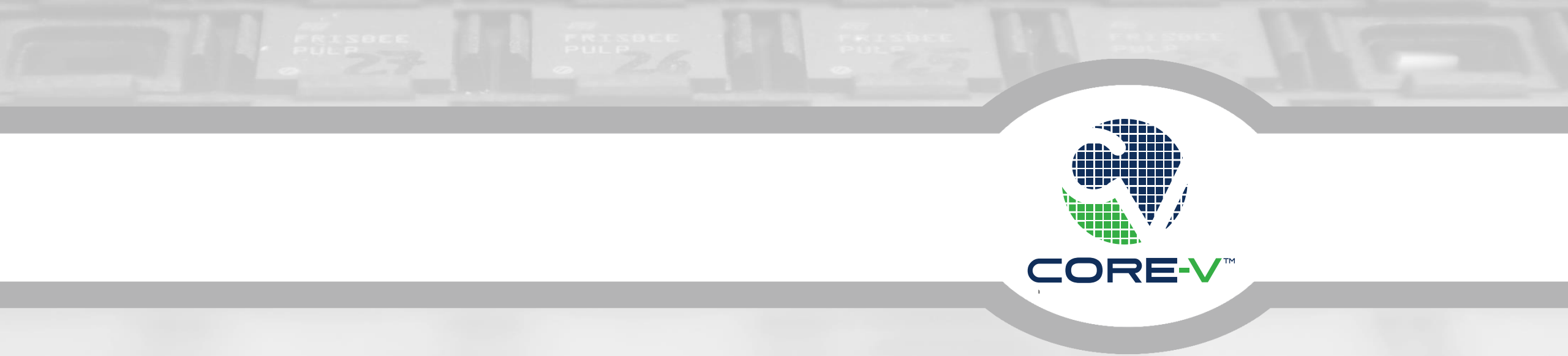
CSR Address [11:10]	[9:8]	[7:4]	Hex	Use and Accessibility
User CSRs				
00	00	XXXX	0x000-0x0FF	Standard read/write
01	00	XXXX	0x400-0x4FF	Standard read/write
10	00	XXXX	0x800-0x8FF	Custom read/write ←
11	00	0XXX	0xC00-0xC7F	Standard read-only
11	00	10XX	0xC80-0xCBF	Standard read-only
11	00	11XX	0xCC0-0xCFF	Custom read-only ←
Supervisor CSRs				
00	01	XXXX	0x100-0x1FF	Standard read/write
01	01	0XXX	0x500-0x57F	Standard read/write
01	01	10XX	0x580-0x5BF	Standard read/write
01	01	11XX	0x5C0-0x5FF	Custom read/write ←
10	01	0XXX	0x900-0x97F	Standard read/write
10	01	10XX	0x980-0x9BF	Standard read/write
10	01	11XX	0x9C0-0x9FF	Custom read/write ←
11	01	0XXX	0xD00-0xD7F	Standard read-only
11	01	10XX	0xD80-0xDBF	Standard read-only
11	01	11XX	0xDC0-0xDFF	Custom read-only ←
Hypervisor CSRs				
00	10	XXXX	0x200-0x2FF	Standard read/write
01	10	0XXX	0x600-0x67F	Standard read/write
01	10	10XX	0x680-0x6BF	Standard read/write
01	10	11XX	0x6C0-0x6FF	Custom read/write ←
10	10	0XXX	0xA00-0xA7F	Standard read/write
10	10	10XX	0xA80-0xABF	Standard read/write
10	10	11XX	0xAC0-0xAFF	Custom read/write ←
11	10	0XXX	0xE00-0xE7F	Standard read-only
11	10	10XX	0xE80-0xEBF	Standard read-only
11	10	11XX	0xEC0-0xEFF	Custom read-only ←
Machine CSRs				
00	11	XXXX	0x300-0x3FF	Standard read/write
01	11	0XXX	0x700-0x77F	Standard read/write
01	11	100X	0x780-0x79F	Standard read/write
01	11	1010	0x7A0-0x7AF	Standard read/write debug CSRs
01	11	1011	0x7B0-0x7BF	Debug-mode-only CSRs
01	11	11XX	0x7C0-0x7FF	Custom read/write ←
10	11	0XXX	0xB00-0xB7F	Standard read/write
10	11	10XX	0xB80-0xBBF	Standard read/write
10	11	11XX	0xBC0-0xBFF	Custom read/write ←
11	11	0XXX	0xF00-0xF7F	Standard read-only
11	11	10XX	0xF80-0xFBF	Standard read-only
11	11	11XX	0xFC0-0xFFF	Custom read-only ←

RISC-V priviledged spec, v20190608, table 2.1

- CVA6 requires a few custom CSRs. How to map them?
- Suggestion for RISC-V custom ranges:
  - First half (lower addresses): reserved for OpenHW specifications and evolutions (us!)
    - To foster upward compatibility in next gen cores
  - Second half (upper addresses): free for users (research, proprietary versions…)
- Example: 0x7C0-0x7FF
  - 0x7C0: I cache control
  - 0x7C1: D cache control
  - 0x7C2-0x7DF: reserved for OpenHW future use
  - 0x7E0-0x7FF: empty space (for user's use, research projects…)

1st half / 2nd half

- Discussion
  - If a custom extension becomes lateran OpenHW specification
    → CSR address update recommended
  - CV32E40S does currently not comply with this recommendation.
  - Common definitions between E4 and A6 series?
    - E.g. If CV32E40❀ gets a D-cache, map its CSR at 0x7C1
    - Need to push commonalities up to the definition of to bit fields?
  - Consider this as a rule or a recommendation?



CV32E40X status

Øystein Knauserud

CV32E40X p.1 (overlaps CV32E40S p.1)

- New controller at same feature level as the old one
  - Missing handling of LSU PMA errors and LSU bus errors
- RVFI port to new controller going well
  - ci_check passing with RVVI enabled
- Further cleanup of valid/ready signalling for all stages and modules
- Performance counter events moved to controller
  - Currently only minstret implemented
- Non-optimized B-extension included (Zba, Zbb, Zbs)
- Multiplier and divider can now be interrupted at any time

Thank you!