Western Digital.

Overlay Manager for RISC-V

A Software paging

Ofer Shinaar

Senior Manager, Firmware & Toolchain
Next Generation Platform Technologies

Agenda

- Introduction: Solving code space limitations with Software
- Basic concepts
- Building blocks: SW and Tool chain blocks
- Summarize

Overlay – Introduction

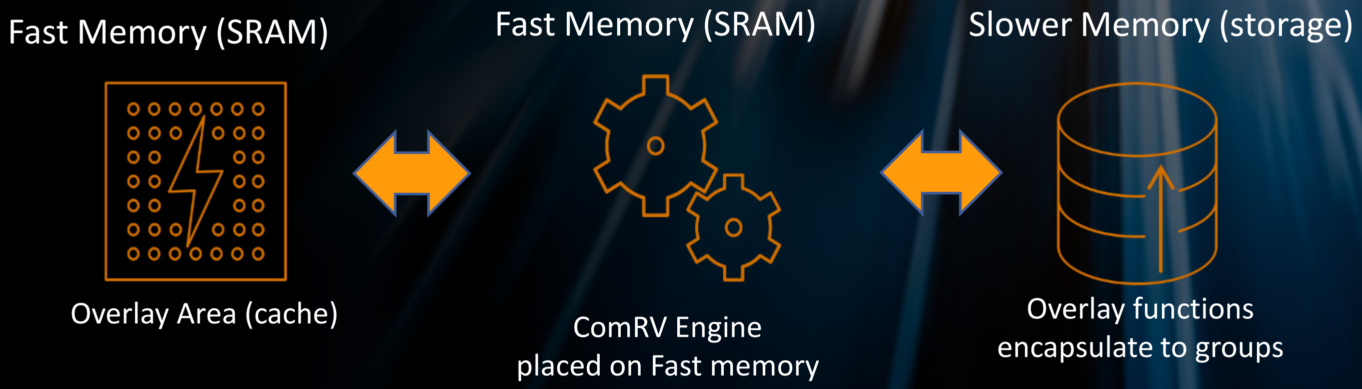
Solving code space limitations with Software

- In the early days of embedded computing there was a technique to load code on Real-Time at the moment it was needed for execution.
- This technique was named Overlay, and it was threaded with the toolchain (compiler, liker, etc.) providing easy application-interface for the SW developers.
- Today, IoT devices (Internet of things) are very strict with memory size and power, alongside requirements for simple HW implementation which does not contain MMU or high-end operation system, to manage it (linux/windows)
- This technique gives the flexibility in reducing physical memory, and therefore reduces energy and space. It saves memory in magnitude proportions over any solution in the ISA domain (eABI, code-density, bitmanip, etc…)

Basic concepts – on overlay engine

Run-Time Module/Engine

- A run-time module operates on the fast memory deciding which function to load or not from a storage device, to fast memory heap
- Code is dynamically loaded to “cache” and executed according the program flow
- The run-time module manages the cache and responsible for invoking the overlay functions



Basic concepts – on overlay engine

User usage

Example **code without** ComRV:

```
void bar(void);
void foo(void)
{
    bar();
}
```

Toolchain generated code:

```
:
jal x1, 0x12345678       ; bar()
:
```

compiler creates
special calls

Example **code with** ComRV:

```
void __attribute__ ((overlaycall)) bar(void);
void foo(void)
{
    bar();
}
```

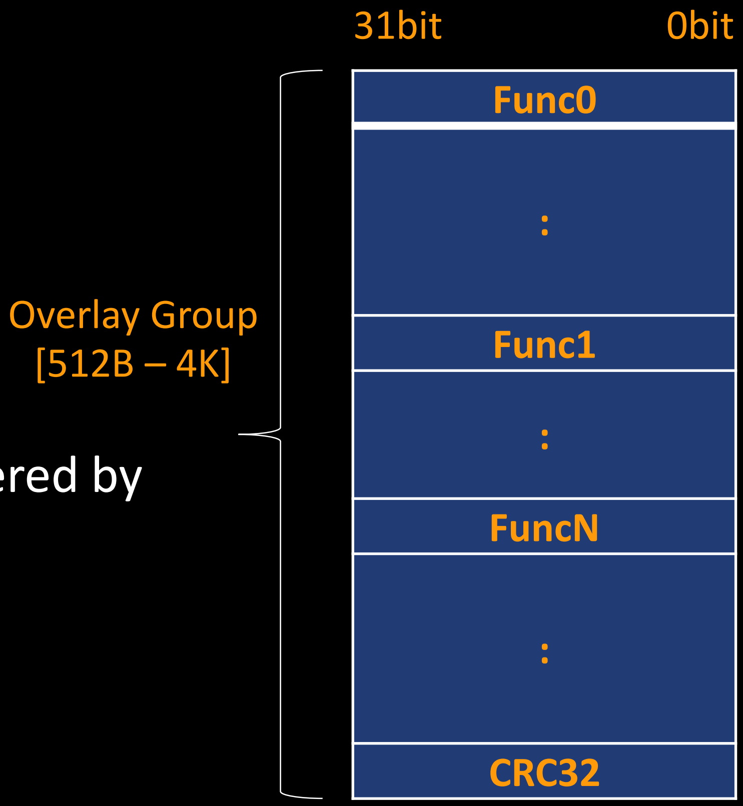
Toolchain will generated code

```
:
li    x31, 0x04C38835   ; bar() "token"
jalr x1, x30            ; ComRV RT-engine
:
```

Overaly – Basic concepts

Functions Group

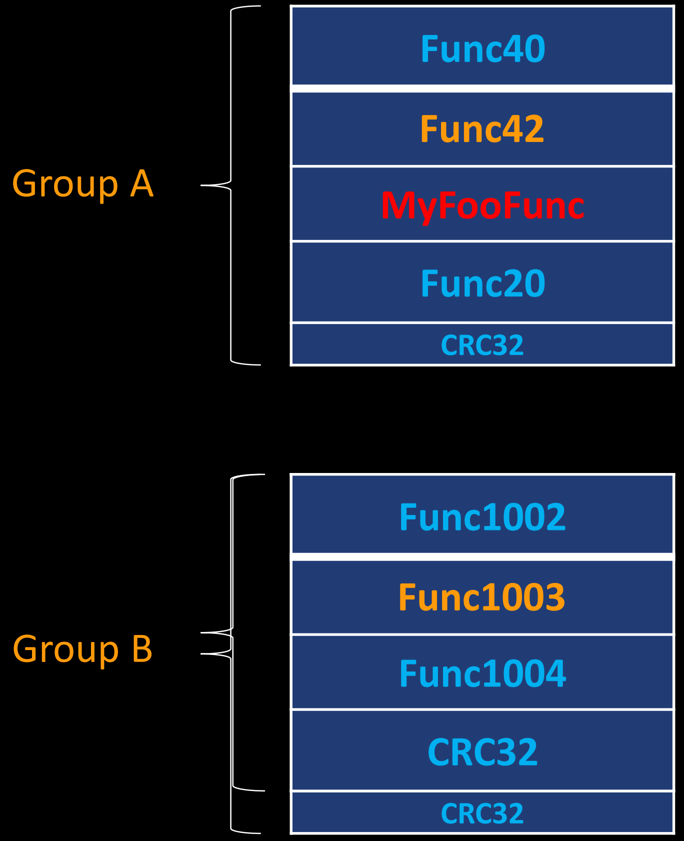
- Overlay group is a container for functions & read-only-data
- When the RT-Module decides to load a function, a full group will be loaded.
- Size of a group is between 512Bytes – 4kBytes
- Gathered functions to groups is done:
  - Manually: User “register” function to group
  - Automatically: Done by external toolchain utility on link-time, triggered by the linker



Overlay – Basic concepts

Multi Grouping (extension feature)

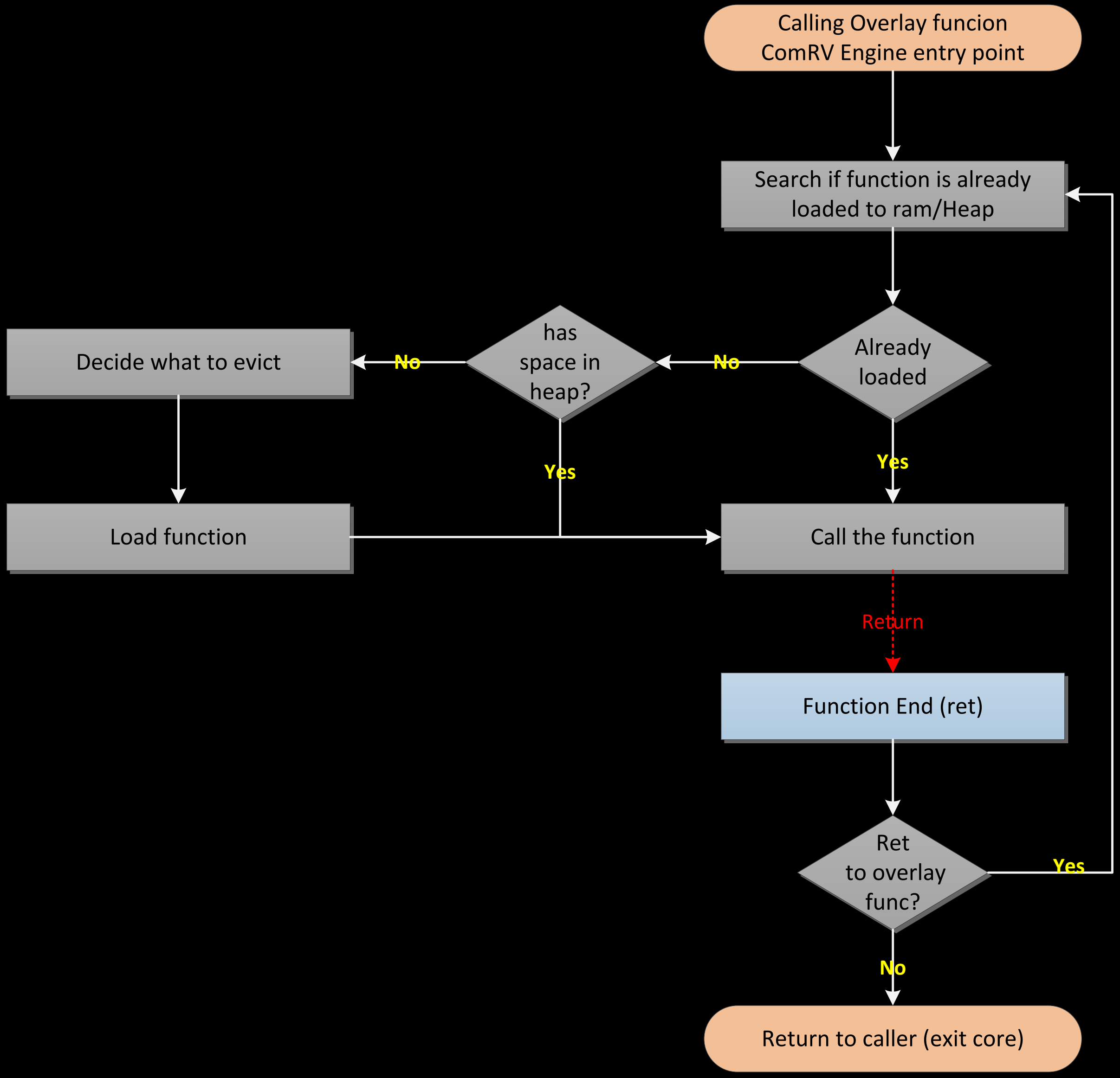
- Sometimes different SW scenarios can run the same function
- Example case study on overlay:
  - We have small cache that only fits one group
  - MyFooFunc is in Group A , it is used by Func42 and also needed by Func1003 which is Group B.
  - Meaning we need to evict A when B is running, and back to A when MyFooFunc is done running.
  - Results = to many loads = to much time
- Multi Groups will gives the option to put MyFooFunc both in Group A and B.



Overlay – Basic concepts

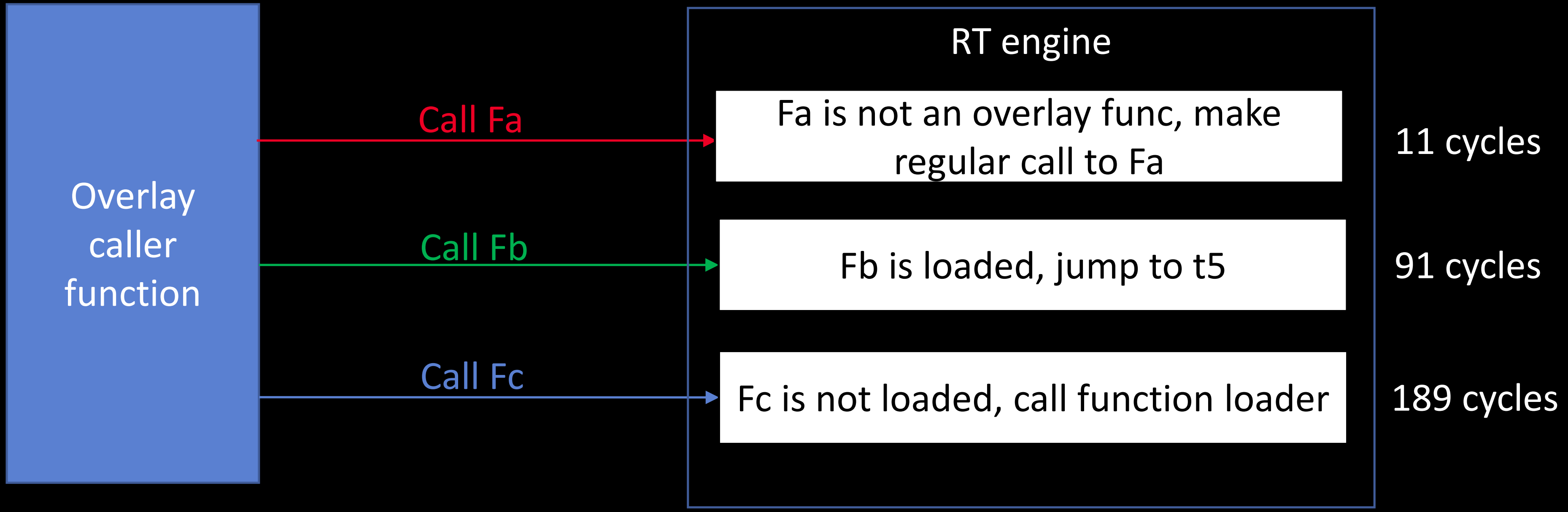
Logic flow

- Each overlay function will be passed through ComRV engine
- The engine is written in C/Asm so it is threaded with RISC-V ISA
- The engine manages the load/evict



Some numbers

ComRV – Cacheable Overlay Manager for RISC-V



Overlay – Building blocks

SW and Tool chain blocks

- For Overlay we need few changes in the toolchain and support utilities
- Compiler:
  - The compiler creates special call for overlay func
- Linker:
  - Create descriptors (tokens) for functions, and offset tables for overlay functions
- Debugger:
  - Provides easy interface to the users for debugging overlay function and overlay core.
- Utilities:
  - Grouping tool, other service utilities…

Summarize

Why we need it for RISC-V ?

- We already know from our partners and customers, that several companies in the 32-MCU domain, are also interested in reviving Overlay.
- Based on our research, something like Overlay manager is not available for RISC-V especially not in open source. We believe that supporting this feature on RISC-V, with RTOS or bare-metal, will significantly improve likelihood of adoption in smaller IoT systems.
- Overlay holds complex RT engine alongside a complex change in the RISC-V toolchains
- This complexity sometime reflects a risk for commercial companies on making the decision to migrating to RISC-V, since there is no overlay solution on RISC-V.
- We believe that having overlay in the risc-v tool-set will reduce this risk.

Summarize - What we can bring in ???...

ComRV – Cacheable Overlay Manager for RISC-V

- WD and Embecosm developed and open sourced the ComRV. The WD solution for Overlay
- ComRV is tested (and keeps on been tested) and proved as a working solution.
- ComRV rises many technical questions and interest by the community and commercial companies , when presenting it on RISCV conventions and Meet-ups.
- We wish to use it as a reference for software standardizing overlay for RISC-V. Pulling in more opinions and demands.
- We commit to finalizing it, documenting it, developing it, and finally deploying it

Western Digital®