Chapter10. Porting Applications from the ARM7 to the Cortex-M3

2019年12月24日 14:29

NOTE TAKING AREA

Low-level code, such as hardware control, task management, and exception handlers, requires most changes.
Application codes changes minor.

System characteristics

Differences: memory map, interrupts, memory protection unit (MPU), system control, and operation modes.

- ARM7's memory and peripherals can be **located in almost any address** (remap vector table to static random access memory, SRAM after boot), while Cortex-M3 has a **predefined** memory map.
  - Memory address differences resolved in compile and linking stage.
  - Peripheral code porting could be more time consuming.
  - Device driver codes might need to be complete rewritten.
- Interrupts: Cortex-M3 sets **PRIMASK** and clear manually, new code to set up interrupt **priority** levels and **vector addresses** for interrupts. FIQ interrupt **doesn't automatic store registers into stack**. NVIC (nested vectored interrupt controller) code can be removed. Various **fault status registers** provided.
- Memory protection unit: **ARM720T** has a memory management unit (MMU), which cannot map to Cortex-M3.
- System control: Cortex-M3 has built-in instructions for entering sleeping mode. Code about **system management features** will need to be rewritten.
- Operation modes: change to exceptions. FIQ can be ported to IRQ.

Assembly language files

Thumb state: most case can be reused. Not supported codes are:

- Tries to switch to ARM state.
- SWI is replaced by SVC, update **parameters passing and return** code.

The program can only access the stack in **full descending** operations.

ARM state: rewrite **vector table**, **register initialization** different. Remove **mode switching and state switching** codes. Interrupt enabling and disabling by setting **PRIMASK and FAULTMASK** register. No **coprocessor accesses** supported.
There is no F bit in Cortex-M3 (no FIQ mode).
Interrupt handler and interrupt return: no vector table->branch to interrupt handler step, return not by manually change program counter but load **EXC_RETURN** into program counter, instructions like **MOVS and SUBS** should not use.
Nested interrupt doesn't need to switch the processor. Save r8-r11 to stack manually.
SWI is replaced by SVC. No SWP instruction in Cortex-M3. cpsr is replaced by xpsr (combined program status register), spsr is removed. Convert conditional execution to IF-THEN (IT).
Pipelined program counter plus 4 instead of 8. Lower 2 bits of r13 is always 0.

Also can compile ARM code into Thumb code, and then move to Cortex-M3.

- Some instructions may be replaced by multiple instructions.
- Long branch range and large immediate data values should be modified to Thumb-2 manually.

C language files

Most case can be recompiled into Cortex-M3. Possible problems:

- Inline assemblers inside _asm keyword.
- Interrupt handler with _irq should be removed.
- Pragma directives $\#pragma\ arm$ or $\#pragma\ thumb$ should be removed.

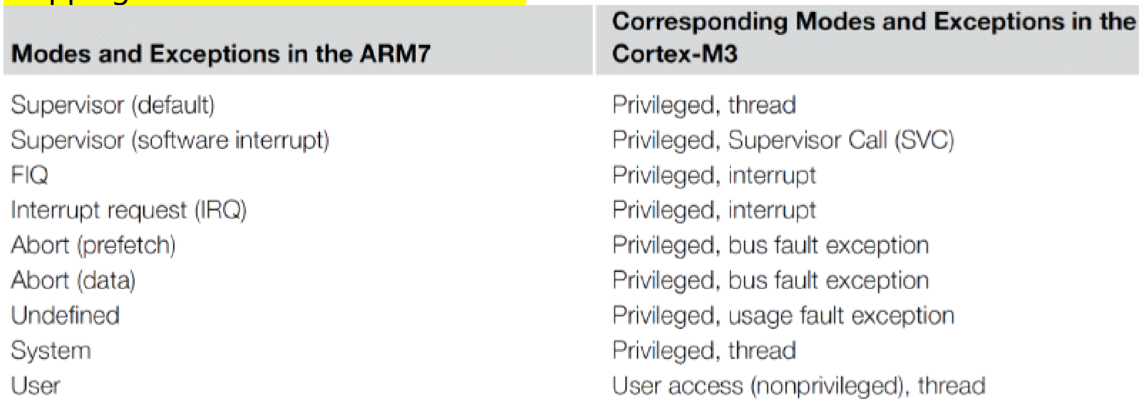
Some precompiled object files cannot be used, recompile using Thumb-2.

Optimization

Areas should be explored: use of the Thumb-2 instruction, bit band, multiply and divide, immediate data, branches, boolean data, bit-field processing, IT instruction block, ARM / Thumb state switching.

CUE COLUMN

Mapping ARM7 modes to Cortex-M3



Modes and Exceptions in the ARM7	Corresponding Modes and Exceptions in the Cortex-M3
Supervisor (default)	Privileged, thread
Supervisor (software interrupt)	Privileged, Supervisor Call (SVC)
FIQ	Privileged, interrupt
Interrupt request (IRQ)	Privileged, interrupt
Abort (prefetch)	Privileged, bus fault exception
Abort (data)	Privileged, bus fault exception
Undefined	Privileged, usage fault exception
System	Privileged, thread
User	User access (nonprivileged), thread

SUMMARIES

1. Porting ARM7 to Cortex-M3: differences and topics.