Proposal for adding debugger on the Aa model

Titto Thomas

October 1, 2015

Current implementation

In ccu.Aa

- A `debug_daemon` module that currently supports only reading register contents and memory. It also supports writing to the registers and memory as well.
- In `ccu_daemon` checks for the processor mode for `PROCESSOR_DEBUG_RESET`.

Proposed additions

In ccu

- Pass the processor mode information (`PROCESSOR_DEBUG_RESET`) to the `teu` blocks for enabling breakpoint and watchpoint checks.
  - replace the current single bit `single_step_mode_selected` with a 2bit `processor_mode_selected`
- When there is a breakpoint / watchpoint hit informed by the `teu`, change the processor mode to single step mode (`PROCESSOR_SINGLE_STEP_RESET`) and let the `teu` finish executing the current instruction.
- Abort further instruction fetch and then transfer the control over to the `ccu_daemon` and wait for a continue signal. If the gdb has detached then let the `teu` continue in normal mode, otherwise continue in debug mode.

In teu

- Store 4 breakpoint and 4 watchpoint registers in the register file.
- If debug mode is enabled then perform a check of calculated memory address with the currently active watchpoints in the `load_store` and pass on the information to `iretire`.
- `iretire` should stop execution and hand over the control to `teu` in case of any watchpoint hit (informed by the `load_store`) or breakpoint hit (by comparing breakpoint registers and current PC) if debug mode is enabled.

In debug_daemon

- Wait for the control transfer from the `ccu` and then communicate with *GDB software server* over pipes. Execute their commands like reading / writing to registers / memory and setting / clearing of watchpoints / breakpoints.
- If the *GDB software server* send a continue message, then inform the `ccu` to continue in debug mode.
- If the *GDB software server* send a detach / kill message, then inform the `ccu` to continue in normal mode.