POLITECNICO DI MILANO

Optimization of the memory architecture

Fabrizio Ferrandi

Politecnico di Milano
Dipartimento di Elettronica, Informazione e Bioingegneria
fabrizio.ferrandi@polimi.it

Activity 3.1 – memory allocation policy

Explore different allocation policy

```
--memory-allocation-policy=<type>
        Set the policy for memory allocation. Possible values for the <type>
        argument are the following:
            ALL_BRAM           - all objects that need to be stored in memory
                                 are allocated on BRAMs (default)
            LSS                - all local variables, static variables and
                                 strings are allocated on BRAMs
            GSS                - all global variables, static variables and
                                 strings are allocated on BRAMs
            NO_BRAM            - all objects that need to be stored in memory
                                 are allocated on an external memory
            EXT_PIPELINED_BRAM - all objects that need to be stored in memory
                                 are allocated on an external pipelined memory
```

Activity 3.1 – Solution Hint


```
$ bambu adpcm.c --memory-allocation-policy=LSS
--clock-period=15 --simulate -v3
```

Look for the log section:

```
Memory allocation information:
    Variable external to the top module: test_result - 25438
- test_result
      Id: 25438
      Base Address: 1073741824
      Size: 400
      Is a Read Only Memory
      Used &(object)
      Number of functions in which is used: 1
      Maximum number of references per function: 1
      Maximum number of loads per function: 1
    …
```

Activity 3.1 - Solution


```
$ bambu adpcm.c --memory-allocation-policy=ALL_BRAM
--clock-period=15 --simulate -v3

$ bambu adpcm.c --memory-allocation-policy=LSS
--clock-period=15 --simulate -v3

$ bambu adpcm.c --memory-allocation-policy=GSS
--clock-period=15 --simulate -v3

$ bambu adpcm.c --memory-allocation-policy=NO_BRAM
--clock-period=15 --simulate -v3

$ bambu adpcm.c
--memory-allocation-policy=EXT_PIPELINED_BRAM
--clock-period=15 --simulate -v3
```

Activity 3.2 – Multi-channel design space exploration


```
--channels-type=<type>
      Set the type of memory connections.
      Possible values for <type> are:
          MEM_ACC_11 - the accesses to the memory have a single direct
                       connection or a single indirect connection (default)
          MEM_ACC_N1 - the accesses to the memory have n parallel direct
                       connections or a single indirect connection
          MEM_ACC_NN - the accesses to the memory have n parallel direct
                       connections or n parallel indirect connections

  --channels-number=<n>
      Define the number of parallel direct or indirect accesses.
```

- When BRAMs are involved only two ports at maximum could be given
- When option `--memory-allocation-policy=EXT_PIPELINED_BRAM` is given the number of channels could be greater than 2

Activity 3.2 – Solution Hint


```
$ bambu adpcm.c --channels-type=MEM_ACC_NN --memory-
allocation-policy=EXT_PIPELINED_BRAM --channels-number=4
--clock-period=15 --simulate -v3
```

Look how long it take the simulation.

Consider `-fwhole-program` option

Activity 3.3 – Control the Load/Store latency


```
--memory-ctrl-type=type
       Define which type of memory controller is used.
       Possible values for the <type> argument are the following:
           D00 - no extra delay (default)
           D10 - 1 clock cycle extra-delay for LOAD, 0 for STORE
           D11 - 1 clock cycle extra-delay for LOAD, 1 for STORE
           D21 - 2 clock cycle extra-delay for LOAD, 1 for STORE

--bram-high-latency=[3,4]
       Assume a 'high latency bram'-'faster clock frequency'
       block RAM memory based architectures:
       3 => LOAD(II=1,L=3) STORE(1).
       4 => LOAD(II=1,L=4) STORE(II=1,L=2).

--mem-delay-read=value
       Define the external memory latency when LOAD are performed (default 2).

--mem-delay-write=value
       Define the external memory latency when LOAD are performed (default 1).
```

Activity 3.3 – Solution Hint


```
$ bambu mips.c --memory-ctrl-type=D21 --channels-
type=MEM_ACC_NN --memory-allocation-
policy=EXT_PIPELINED_BRAM --channels-number=4
--clock-period=15 --simulate -v3
```

Look how long it take the simulation.

```
$ bambu mips.c --bram-high-latency=4 --channels-
type=MEM_ACC_NN --clock-period=15 --simulate -v3
```

Look how long it take the simulation.

Activity 3.4 – Control asynchronous memories inference


```
--do-not-use-asynchronous-memories
        Do not add asynchronous memories to the possible set
        of memories used by bambu during the memory allocation
        step.

--distram-threshold=value
        Define the threshold in bitsize used to infer
        DISTRIBUTED/ASYNCHRONOUS RAMs (default 256).
```

Activity 3.4 – Solution Hint


```
$ bambu mips.c --do-not-use-asynchronous-memories -fwhole-program --clock-period=15 --simulate -v3
```

Look how long it take the simulation.

```
$ bambu mips.c --distram-threshold=1024 -fwhole-program --clock-period=15 --simulate -v3
```

Look how long it take the simulation.

Activity 3.5 – Alignment hints


```
--unaligned-access
        Use only memories supporting unaligned accesses.

--aligned-access
        Assume that all accesses are aligned and so only
        memories supporting aligned accesses are used.
```

Activity 3.5 – Solution Hint


```
$ bambu mips.c --unaligned-access -fwhole-program --clock-period=15 --simulate -v3
```

Look how long it take the simulation.

```
$ bambu mips.c --aligned-access -fwhole-program --clock-period=15 --simulate -v3
```

Look how long it take the simulation.

Activity 3.6 – customize memory layout


```
--base-address=address
     Define the starting address for objects allocated externally to the top
     module.

--initial-internal-address=address
     Define the starting address for the objects allocated internally to the
     top module.
```

Activity 3.5 – Solution Hint


```
$ bambu mips.c --base-address=1024 --memory-allocation-
policy=LSS --clock-period=15 --simulate -v3
```

Look how long it take the simulation.

```
$ bambu mips.c --initial-internal-address=0 --base-
address=1024 --memory-allocation-policy=LSS --clock-
period=15 --simulate -v3
```

Look how long it take the simulation.

Other options


```
--sparse-memory[=on/off]
        Control how the memory allocation happens.
            on - allocate the data in addresses which reduce the decoding logic (default)
           off - allocate the data in a contiguous addresses.

--serialize-memory-accesses
        Serialize the memory accesses using the GCC virtual use-def chains
        without taking into account any alias analysis information.

--do-not-chain-memories
        When enabled LOADs and STOREs will not be chained with other
        operations.

--rom-duplication
        Assume that read-only memories can be duplicated in case timing requires.


--do-not-expose-globals
        All global variables are considered local to the compilation units.

--data-bus-bitsize=<bitsize>
        Set the bitsize of the external data bus.

--addr-bus-bitsize=<bitsize>
        Set the bitsize of the external address bus.
```