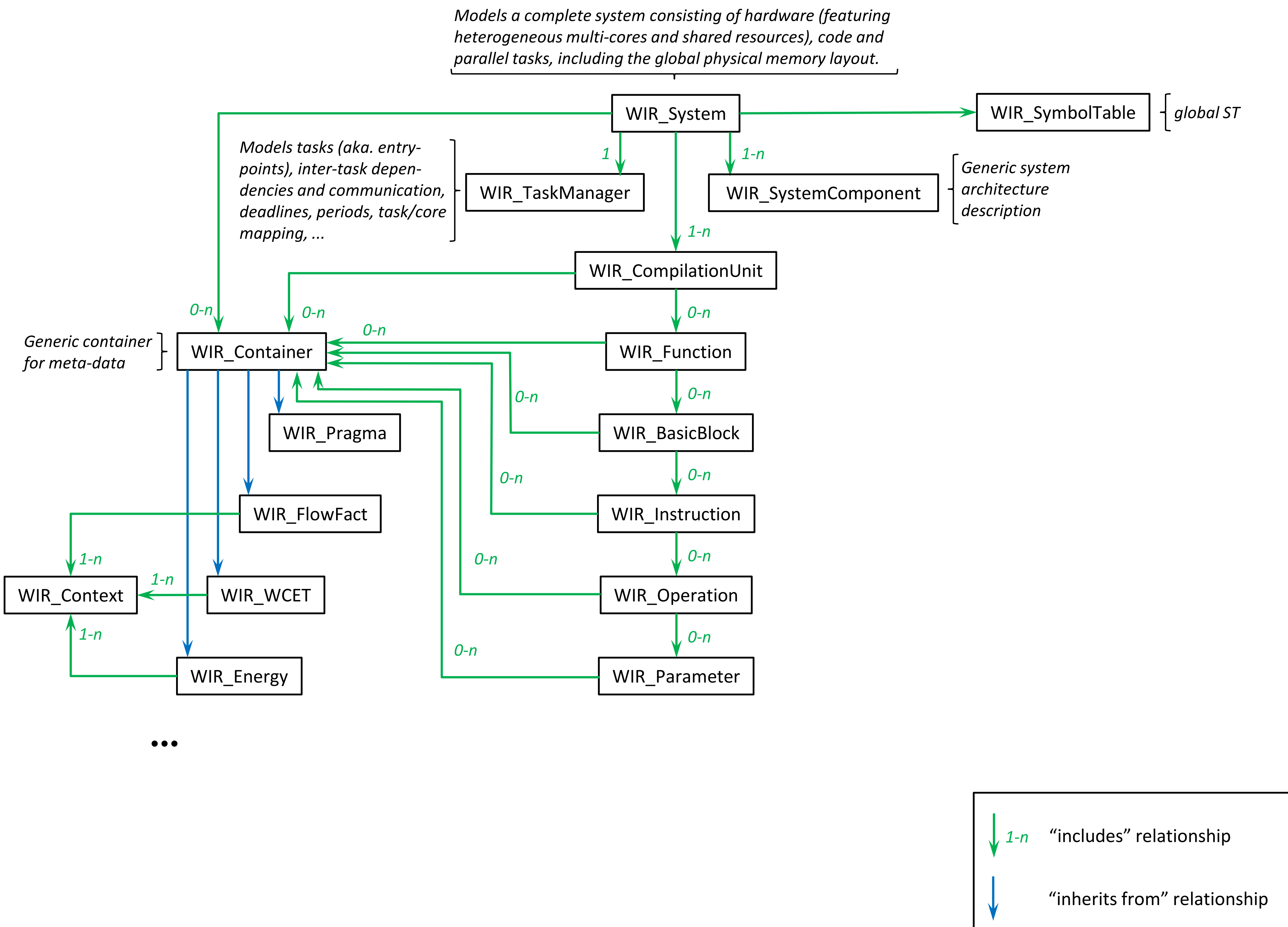
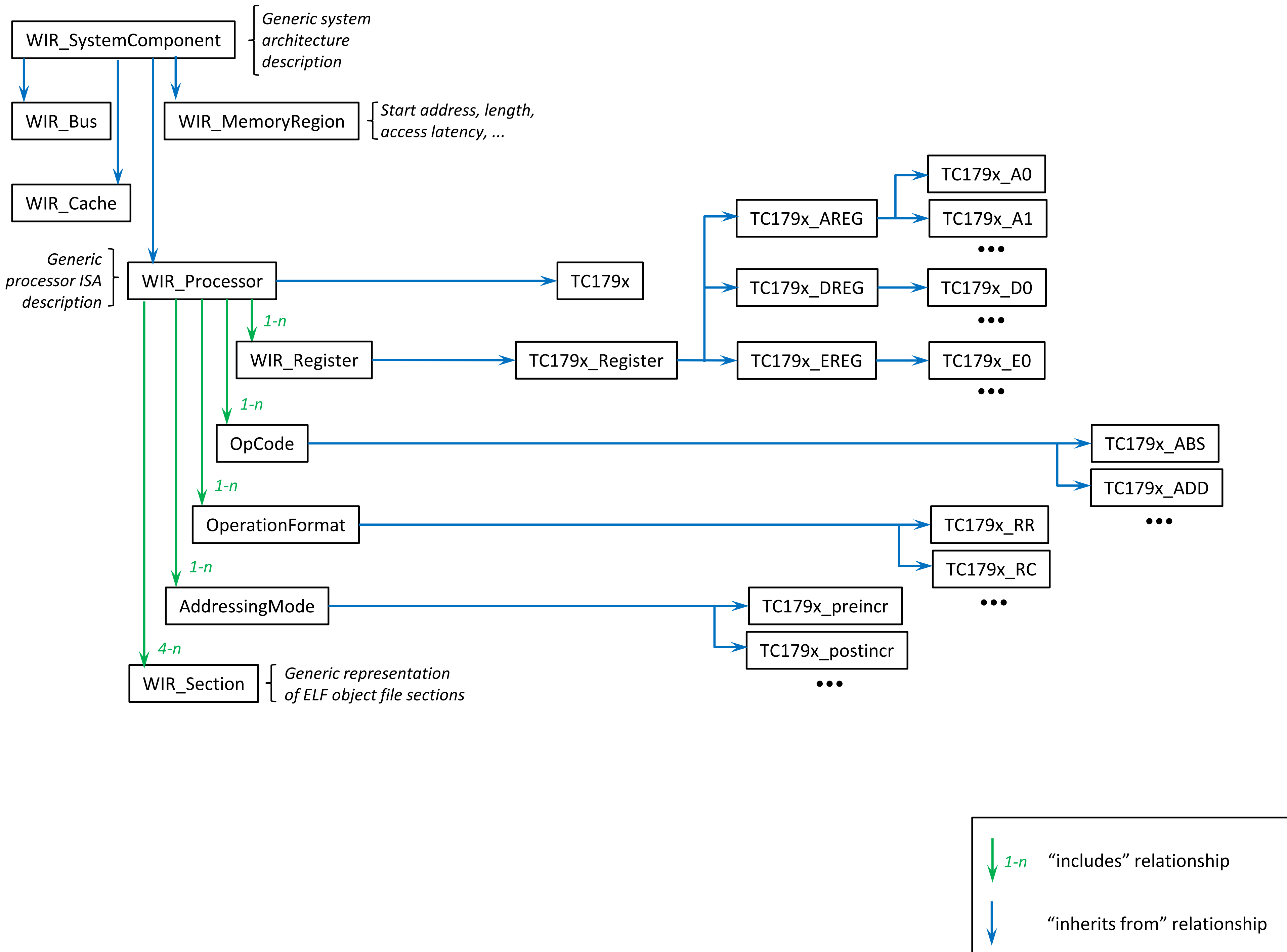

Models a complete system consisting of hardware (featuring heterogeneous multi-cores and shared resources), code and parallel tasks, including the global physical memory layout.
WIR_System
WIR_SymbolTable
global ST
Models tasks (aka. entry-points), inter-task dependencies and communication, deadlines, periods, task/core mapping, ...
1
WIR_TaskManager
1-n
WIR_SystemComponent
Generic system architecture description
1-n
WIR_CompilationUnit
0-n
0-n
0-n
Generic container for meta-data
WIR_Container
0-n
WIR_Function
0-n
0-n
WIR_Pragma
WIR_BasicBlock
0-n
0-n
WIR_FlowFact
WIR_Instruction
1-n
0-n
0-n
WIR_Context
1-n
WIR_WCET
WIR_Operation
1-n
0-n
0-n
WIR_Energy
WIR_Parameter
•••
1-n "includes" relationship
"inherits from" relationship


WIR_SystemComponent
Generic system architecture description
WIR_Bus
WIR_MemoryRegion
Start address, length, access latency, ...
WIR_Cache
Generic processor ISA description
WIR_Processor
TC179x
1-n
WIR_Register
TC179x_Register
TC179x_AREG
TC179x_A0
TC179x_A1
TC179x_DREG
TC179x_D0
TC179x_EREG
TC179x_E0
1-n
OpCode
TC179x_ABS
TC179x_ADD
1-n
OperationFormat
TC179x_RR
TC179x_RC
1-n
AddressingMode
TC179x_preincr
TC179x_postincr
4-n
WIR_Section
Generic representation of ELF object file sections
1-n
"includes" relationship
"inherits from" relationship

Open questions:

- *What about other aiT/CRL-specific things beyond WIR_Context?*

Details omitted here for the sake of simplicity of the diagrams:

- *Bus arbitration policies*
- *Distinction between loop-bounds and flow-restrictions below WIR_FlowFact*
- *All core classes (i.e., from WIR_System down to WIR_Parameter) inherit from one base class so that all WIRs, functions, BBs, etc. obtain a unique numerical identifier that can be queried. This numerical identifier should then be used to implement sets of operations and so on.*
- *Solely use of STL containers, Boost serialization, maybe Boost graphs.*
- *Proper copy constructors right from the beginning. Beware: copying must preserve the order/monotonicity of the numerical IDs mentioned above in order to keep copied STL sets in exactly the same order.*
- *Simple and clean API with minimized exposure of pointers to WIR objects. C++ references shall be used wherever possible. 64bit-safe design right from the beginning!* **`const`***ness of methods and arguments shall be annotated wherever possible.*
- *Testbench featuring unit tests so that classes and their methods are explicitly tested.*
- *ASM code parsers for different processor architectures.*