

C/C++にて迅速・容易にハードウェア設計可能な 高位設計/ASIP設計ツール登場

C2RGate

C/C++記述で迅速・容易にハードウェア設計を行いRTL生成。
豊富なライブラリにより、ASIP機能も充実。
画像処理、セキュリティ、AI、IoTチップ等
様々な分野で使用可能です。



C2R Gate オプション販売予定

C2R Idea Notes

例) MPU (RISC-V) モデル、バスモデル等

C2RTL技術とは

C2RTL技術は、C/C++記述ベースのRTL (Register-Transfer Level) 設計フレームワークです。

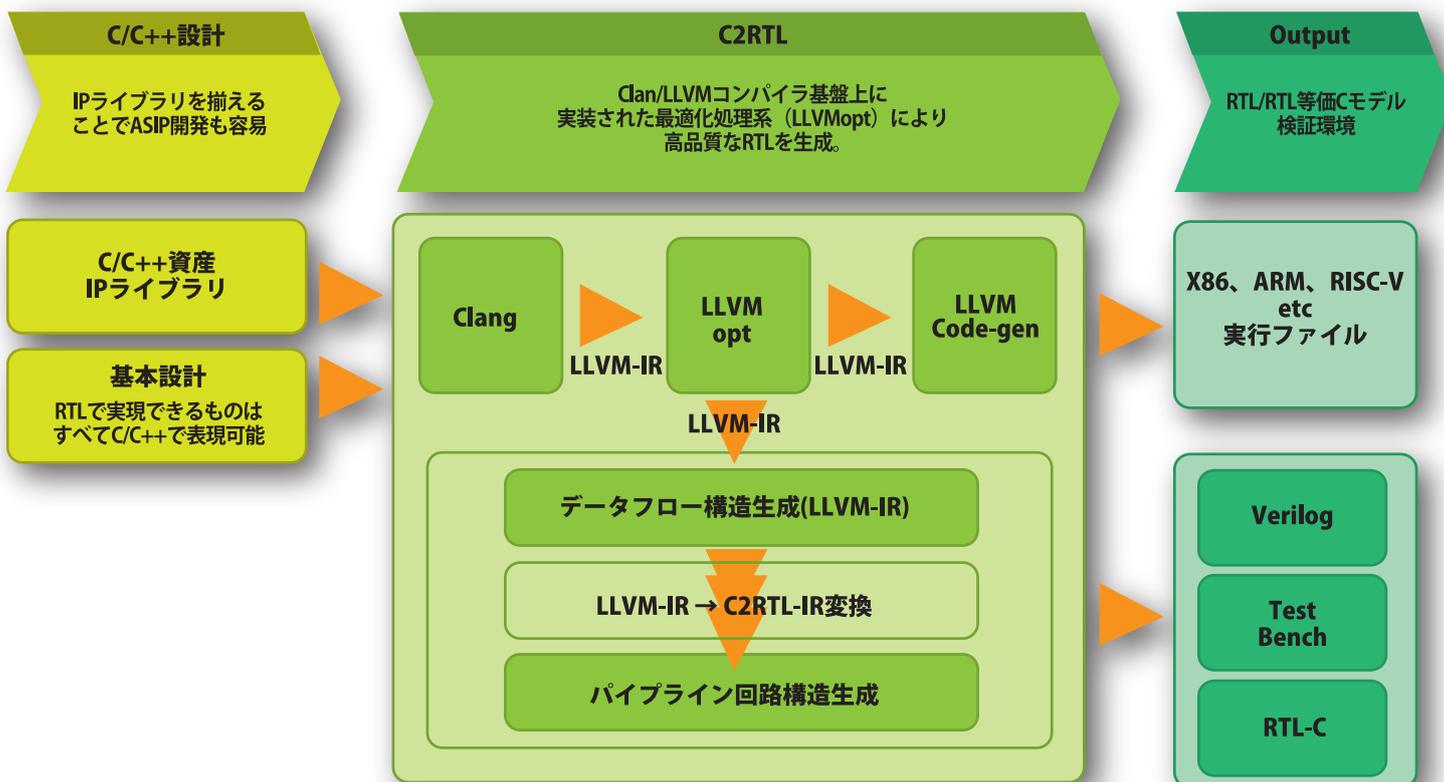
LLVMコンパイラ基盤を利用しLLVM中間言語 (LLVM-IR) を入力とした最適化処理系 (LLVM optimizer : opt) エンジンとして、LLVMコンパイラ基盤上に実装されています。

任意のハードウェア回路構造をソフトウェア記述 (C/C++言語) によって直接表現できます。ソフトウェア記述そのものがRTL構造を表現する点が特徴です。

C2R Gateについて

C2R Gateは、C2RTL技術をベースとした高位設計・ASIP設計ツールです。IP (プロセッサ、バス、IO等) ライブラリにより、容易に高位設計・ASIP設計が行えます。

SoCを構成する全てのIP部品と、SoCインテグレーション(各IPの接続等)を全て同一ソフトウェア言語 (C/C++) で設計出来るので、可読性が高く且つ開発工数を削減することができます。また、高速システム検証モデル (動作レベル、サイクルレベル) とRTL回路モデルを同時に生成可能になり、SoC開発効率を向上することができます。



C2R Gateの特徴 1

システム設計の「見える化」が可能

- ・ソフトウェア記述 (C/C++) でシステム全体の動作検証モデル・論理合成モデルを统一的に表現
- ・高い抽象度で全システムを見通すことが可能
- ・ソフトウェア記述によるハードウェアIP開発の容易化
- ・ソフトウェア記述によるシステム検証が可能

C2R Gateの特徴 2

C/C++記述によるRTL構造表現 (データベース、FSM、サブシステム) が可能

- ・データフロー C/C++記述 (1サイクル記述)
- ・C/C++準拠 (既存C/C++開発環境利用可)
- ・ハードウェア属性記述 (pragma/GCC-attribute) にて Bit幅/レジスタ/メモリを表現

統合化された設計環境を提供

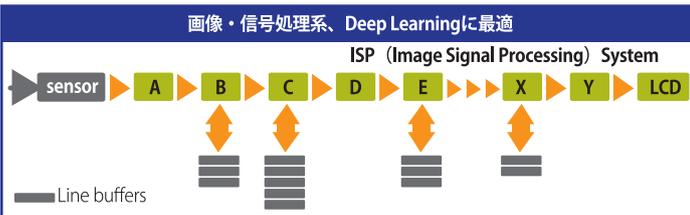
- ・RTL等価Cモデルを自動生成することで元のC/C++開発環境でRTL検証可能
- ・RTL検証環境を自動生成することで検証時間を短縮

C2R Gateの特徴 3

アプリケーションに合わせたC/C++データフロー記述が可能

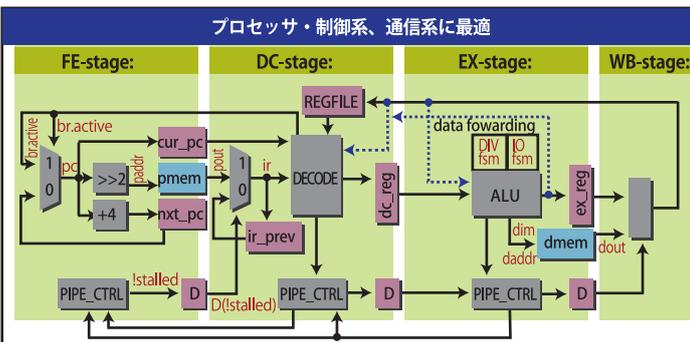
<データフロー型単純パイプライン記述>

- ・高並列/パイプライン処理
- ・パイプラインストール制御なし
- ・パイプライン段数調整/論理遅延平滑化



<フロー制御付きパイプライン記述>

- ・複雑なフロー制御を「逆方向参照」で詳細記述
- ・プロセッサCモデルで必須 (ストール制御、data forwarding)
- ・詳細なパイプライン構造をC/C++データフロー記述で表現可能



動作環境 マシンスペック OS : Windows10 メモリ : 8Gbyte 以上 CPU : Core i5
ソフトウェア環境 MinGW-w64, GnuWin32, icarus Verilog

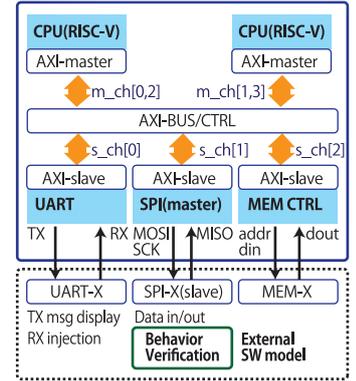
C2R Gateの特徴 4

ASIP開発ツール機能

- ・IPライブラリを準備することで、ASIP開発ツールとして使用可能

例) RISC-Vプロセッサ開発例

IPライブラリ
RISC-V x2 : キャッシュ付き AXI-Master x2/コア
UART : AXI-Slave
SPI : AXI-Slave
MEMCTRL : AXI-Slave
AXI-BUS : 4-Master, 3-Slave



```
SoC階層属性
_C2R_MODULE_ #define C2R_MODULE __attribute__((C2R_module))
int RVProcAXI(D_FIFO_PORT *in_fifo, D_FIFO_PORT *out_fifo,
D_FIFO_PORT *in_fifo2, D_FIFO_PORT *out_fifo2,
MEMCTRLPin *mpin, UARTPin *uart, SPIPin *spi, AXI4L::BUS<4,3> *axib) {
axi_uart.step (&axib->s_ch[0], uart);
axi_spim.step (&axib->s_ch[1], spi);
axi_memctl.step (&axib->s_ch[2], mpin);
int val = cpu1.step (in_fifo, out_fifo, &axib->m_ch[0], &axib->m_ch[2]);
val &= cpu2.step (in_fifo2, out_fifo2, &axib->m_ch[1], &axib->m_ch[3]);
axi_bus_ctrl.connectChannel (axib);
return val;
}
SoCレベル記述: IP トップ関数の呼出し → IP接続記述
```

```
FIFO_PORT in_fifo[2], out_fifo[2];
MEMCTRLPin mpin;
UARTPin uart;
SPIPin spi;
AXI4L::BUS<4, 3> axi_bus = { 0 };
int main (int argc, char *argv[]) {
// initialization codes here...
RV-SoCテストベンチ記述
while (!RVProcAXI (&in_fifo[0], &out_fifo[0], &in_fifo[1], &out_fifo[1], &mpin, &uart, &spi, &axi_bus)) {
spi_ext_slave (&spi);
uart.rx = uart_ext(uart.tx);
mem_ext.update (&mpin);
}
外部ソフトウェアモデル
```

ユーザ定義クラスのみ (Built-inクラスなし)

```
IP階層属性 (個別RTL合成関数)
#define C2R_FUNC(N) __attribute__((C2R_function(N)))
_C2R_FUNC(5)
CPU top: 5段パイプライン
int CPU::step (D_FIFO_PORT * in_fifo, D_FIFO_PORT * out_fifo,
AXI4L::CH *axi_d, AXI4L::CH *axi_i) {
fetch(); decode(axi_i); execute(axi_d); data_mem(); writeback();
return (cpu.halted == 1);
}
D-cache実装のため、1段追加
```

```
_C2R_FUNC(1)
void UART_AXI4L::step (AXI4L::CH *axi,
UARTPin *uart_pin) {
uart.set_pin(uart_pin);
fsm(axi);
}
AXI-UART top
AXI-BUS_CTRL top
_C2R_FUNC(1) template <int MC, int SC>
void AXI4L::CTRL<MC,SC>::
connectChannel(BUS<MC,SC> *bus){ ... }
AXIバス制御
_C2R_FUNC(1)
void MEMCTL_AXI4L::step (AXI4L::CH *axi,
D_MEMCTRLPin *mem_pin) {
din = mem_pin->dout;
mpin.set_outpin(mem_pin);
fsm(axi);
}
AXI-MEMCTL top
_C2R_FUNC(1)
void SPIM_AXI4L::step(AXI4L::CH *axi,
SPIPin *spi_pin) {
spim.set_pin(spi_pin);
fsm(axi);
}
AXI-SPI top
```

問い合わせ先

info-eureka@eureka-oss.com
URL : http://www.eureka-oss.com/

株式会社ユーリカ Eureka!

本社
〒461-0001 愛知県名古屋市中区泉一丁目21-27 泉ファーストスクエア6F TEL : 052-962-3207
新横浜オフィス
〒222-0033 神奈川県横浜市港北区新横浜2-2-8 新横浜ナラビル6F TEL : 045-620-8381

* C2RTLは東京工業大学と一般財団法人新システムビジョン研究開発機構 (NSV財団) で共同開発された大規模回路の高位設計技術です。
* 記載されている企業名・製品名は各社の登録商標または商標です。