FPGAによる数値計算の高速化

理化学研究所 RIKEN
戎崎計算宇宙物理 濱田剛
supported by Information-technology Promotion Agency, Japan IPA

これまでのFPGAを使った数値計算用ハードウェア

浮動小数点演算器の回路規模が大きすぎて非効率

単純なアルゴリズムでもプログラミングが非常に面倒

実装効率, 作業効率に優れたFPGA用コンパイラ

PGPG：Pipeline Generator for Programmable GRAPE

http://progrape.jp/

PGPGコンパイラの基本概念

簡単なデータフロー記述から,

(1) HDL(回路構成情報)

(2) Emulator(回路検証プログラム)

(3) プログラミングインタフェース

の3つを自動生成

PGPGコンパイラによるアルゴリズム記述例

GRAPE (重力加速度計算専用LSI)相当な回路をPGPGで設計した場合の記述

```
/* ------------------------------------------------------------ MACRO */
#define NFLO 26      /* Bit-Length for floating-point word */
#define NMAN 16      /* Bit-Length for mantissa            */
#define NST_SUB   4  /* Pipelining Stages for subtractor   */
#define NST_MULT  2  /* Pipelining Stages for multiplier   */
#define NST_RECI  3  /* Pipelining Stages for recipro      */
#define NST_SQRT  3  /* Pipelining Stages for squer root   */
#define NST_ACCM  4  /* Pipelining Stages for accumlator   */

/* ------------------------- REGISTER VARIABLES, MEMORY VARIABLES */
/IPSET xi[3], ix[][], float, NFLO, NMAN;
/IPSET e2,     eps2,  float, NFLO, NMAN;
/JPSET xj[3], jx[][], float, NFLO, NMAN;
/JPSET mj,    jm[],   float, NFLO, NMAN;
/FOSET sx[3], ax[][], float, NFLO, NMAN;

/* ------------------------------------------------ NUMBER OF PE */
/NVMP  2; /* Number of Virtual Multiple Pipelines */
/NPIPE 5; /* Number of Real Pipelines */

/* ------------------------------------------- PIPELINE DATA FLOW */
pg_float_sub(xi,xj,dx,             NFLO,NMAN,NST_SUB);  // dx[] = xi[] - xj[]
pg_float_mult(dx,dx,dx2,           NFLO,NMAN,NST_MULT); // dx2[] = dx[] * dx[]
pg_float_add(dx2[0],dx2[1],x2y2,   NFLO,NMAN,NST_SUB);  // x2y2 = dx2[0]+dx2[1]
pg_float_add(dx2[2],e2,z2e2,       NFLO,NMAN,NST_SUB);  // z2e2 = dx2[2]+e2
pg_float_add(x2y2,z2e2,r2,         NFLO,NMAN,NST_SUB);  // r2 = x2y2+z2e2
pg_float_sqrt(r2,r1,               NFLO,NMAN,NST_SQRT); // r1 = sqrt(r2)
pg_float_recipro(r1,r1i ,          NFLO,NMAN,NST_RECI); // r1i = 1/r1
pg_float_mult(r1i,r1i,r2i,         NFLO,NMAN,NST_MULT); // r2i = r1i * r1i
pg_float_mult(r1i,r2i,r3i,         NFLO,NMAN,NST_MULT); // r3i = r1i * r2i
pg_float_mult(r3i,mj,mr,           NFLO,NMAN,NST_MULT); // mr = r3i * mj
pg_float_mult(dx,mr,fx,            NFLO,NMAN,NST_MULT); // fx[] = dx[]* mr
pg_float_accum(fx,sx,              NFLO,NMAN,NST_ACCM); // sx[] += fx[]
```

Virtex-IIPro FPGAでの合成例

浮動小数点乗算
(Nearest Even丸め)
35 Slices
1 MULT18x18
137 MHz

浮動小数点平方根
116 Slices
3 MULT18x18
128 MHz

以上約30行程度の記述でプログラミング作業が完了する