A Concurrent Membrane Computing Scheme

Hao Bai* Ph.D, Keyi Shen† and Arko Dey†

Electronic & Computing Engineering Department, Zhejiang University - University of Illinois at Urbana-Champaign, Haizhou East Road, Haining, 314400, Zhejiang, China.

*Corresponding author(s). E-mail(s): Haob.19@intl.zju.edu.cn;
Contributing authors: Keyi.19@intl.zju.edu.cn; Arko.18@intl.zju.edu.cn;
†These authors contributed equally to this work.

Abstract

This part will be finished soon.

Keywords: Chipyard, Gemmini, Transformer, System on Chip, Accelerator, Co-processor

1 Introduction

2 Acknowledgements

We need to sincerely thank Professor K.D. Schewe.

References

Appendix A Code Space

A.1 The code for transformer multiplication and calculation of matrix Q,K,V.

```
for (int count = 0; count < n_head; count++)
  {
    tiled_matmul_auto(wordNum, weightDim, wordDim,
                      (elem_t *)word_vector,
                      (elem_t *)q_mats[count], NULL,
                      (elem_t *)z_qs[count],
                      wordDim, weightDim,
                      weightDim, weightDim,
                      MVIN_SCALE_IDENTITY,
                      MVIN_SCALE_IDENTITY,
                      MVIN_SCALE_IDENTITY,
                      NO_ACTIVATION, ACC_SCALE_IDENTITY,
                      0, false,
                      false, false,
                      false, false,
                      3,
                      accel_type);
    tiled_matmul_auto(wordNum, weightDim, wordDim,
                      (elem_t *)word_vector,
                      (elem_t *)k_mats[count], NULL,
                      (elem_t *)z_ks[count],
                      wordDim, weightDim,
                      weightDim, weightDim,
                      MVIN_SCALE_IDENTITY,
                      MVIN_SCALE_IDENTITY,
                      MVIN_SCALE_IDENTITY,
                      NO_ACTIVATION, ACC_SCALE_IDENTITY,
                      0, false,
                      false, false,
                      false, false,
                      3,
                      accel_type);
    tiled_matmul_auto(wordNum, weightDim, wordDim,
                      (elem_t *)word_vector,
                      (elem_t *)v_mats[count], NULL,
                      (elem_t *)z_vs[count],
                      wordDim, weightDim,
                      weightDim, weightDim,
                      MVIN_SCALE_IDENTITY,
                      MVIN_SCALE_IDENTITY,
                      MVIN_SCALE_IDENTITY,
                      NO_ACTIVATION,
                      ACC_SCALE_IDENTITY,
                      0, false,
                      false, false,
```

```
                        false, false,
                        3,
                        accel_type);
    }
```

A.2 The code for the exponential function mentioned in the Softmax calculation part.

```
float my_exp(elem_t number)
{
  // float exp;
  // a is the slope
  float slope[24] = {0.006737946999085467, 0.0717489450711988,
  0.13736897328580006, 0.20417102465724862, 0.27276942684711947,
  0.34384344502557984, 0.41816603105197925, 0.49664045641486215,
  0.580348760164344, 0.6706178960296387, 0.7691126983842523,
  0.8779702723670761, 1.0000000000000004, 1.1389907283579461,
  1.3001995703630875, 1.4911621147011624, 1.7231018116017323,
  2.0135290773908743, 2.39139462735484, 2.9083003165164474,
  3.666100015528776, 4.897854637692817, 7.279664221697798,
  13.937487150614775};
  // b is the inter
  float x0[24] = {-5, -3.38975777788707, -2.274768975664783,
  -1.773836174919944, -1.4369560632507996, -1.1788714089484582,
  -0.9665265685144582, -0.7834156082000947, -0.6199855288515437,
  -0.4701006376974108, -0.3294748672686874, -0.19487414204749406,
  -0.06366558341604925, 0.0664754970873836, 0.19778114058929605,
  0.3325900427376481, 0.4735679272763055, 0.6240101415956079,
  0.7883219072584459, 0.9728773115664778, 1.187761027381711,
  1.4508594080418584, 1.7998227092372945, 2.3442268793669405};
  float y0[24] = {0.3860500834856052, 0.017587673747372003,
  0.0975869440730206, 0.1663995685965194, 0.23518072618265692,
  0.30557832941474355, 0.37859171088297716, 0.45516249439975104,
  0.5363284835993326, 0.6233139943480013, 0.7176201526404586,
  0.821143279619808, 0.936340493578349, 1.066481574081782,
  1.216037484611534, 1.391315961265941, 1.6015368417007017,
  1.8607640937356675, 2.191610611655227, 2.632955413966755,
  3.257901792686428, 4.22244677011019, 5.931618293254853,
  9.894697852690719};
  for (int i = 0; i < 24; i++)
  {
    if (number < x0[i])
    {
      if (i != 0)
      {
        return slope[i - 1] * (number - x0[i - 1]) + y0[i - 1];
      }
      else
      {
```

```
        return slope[0] * (number - x0[0]) + y0[0];
      }
    }
  }
  return slope[23] * (number - x0[23]) + y0[23];
}
```

A.3 The code for the square root function mentioned in the layer normalization calculation part.

```
float my_sqrt(elem_t number)
{
  float sqrt;
  // a is the slope
  float a[16] = {0.3549, 0.1470, 0.1128, 0.0951, 0.0838,
  0.0758, 0.0697, 0.0648, 0.0609, 0.0576, 0.0548, 0.0523,
  0.0502, 0.0483, 0.0466, 0.0451};
  // b is the inter
  float b[16] = {0, 2.8174, 3.9843, 4.8798, 5.6347, 6.2998,
  6.9011, 7.4540, 7.9687, 8.4521, 8.9093, 9.3441, 9.7596,
  10.1581, 10.5416, 10.9116};
  float step = 7.9438;
  for (int i = 0; i < 16; i++)
  {
    if ((step * i <= number) && (number < step * (i + 1)))
    {
      sqrt = a[i] * number + b[i];
      break;
    }
  }
}
```

Appendix B Big Graph Space

	Gap9	Our Chip
Processing Power	150.8GOPS (32.2GMACs ML & 15.6GOPs DSP)	百GOPS
Power Efficiency	0.33mW/GOP	< 1 W
L1 Memory	128kB	128kB + banked (16 way +)
L2 Memory (RAM)	1.5MB （应为1.6MB）	3MB +
L3 Memory (NVM)	2MB	不需要
External Memory	2x QSPI/OCTO-SPI/HyperBus/SDIO	需要
FC Frequency	400MHz	FPU removed
Cluster Frequency	400MHz (8 RISC(parallel) + 1RISC(master) + 1 APU(Float) + 1 HWPE(Vega: 3 * conv3 engine(3*3 PEs)))	HWPE -> 16*16 PEs WS @ over 500MHz(Area) Or 16*16 PEs WS @ 200MHz(power) * N cluster APU -> LUT based activation 8RISC -> 8 wide SIMD ? 1RISC master(optional)
Fix Point Precision	8,16,32,64-bit	4，8，16
Floating Point Precision	16, 32	V0 不需要
Sound Interface	3 master/slave SAI full duplex, I2S and TDM 4/8/16 ch capable	不需要
Camera Interface	8-bit CPI, 2-lane CSI-2	需要
Package type	WL-CSP 3.7mmx3.7mm - BGA 5.5mm x 5.5mm 制程工艺scale后等效28nm面积（WL-CSP）：4.7mm*4.7mm	Area < 4*4
Process	22 nm	28 nm

Fig. B1 The comparison of *Gap 9* and our chip.

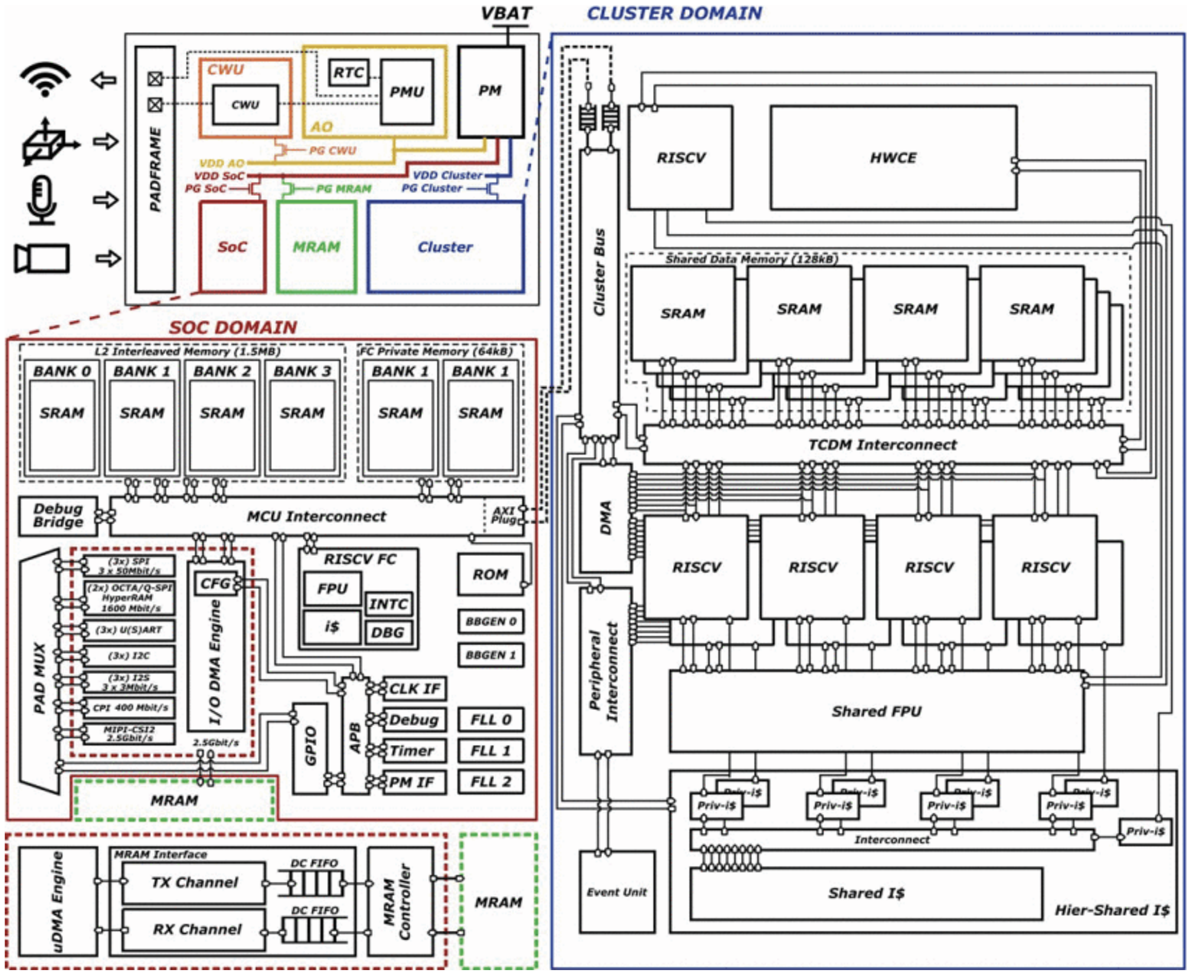

Fig. B2 The original SoC Layout.

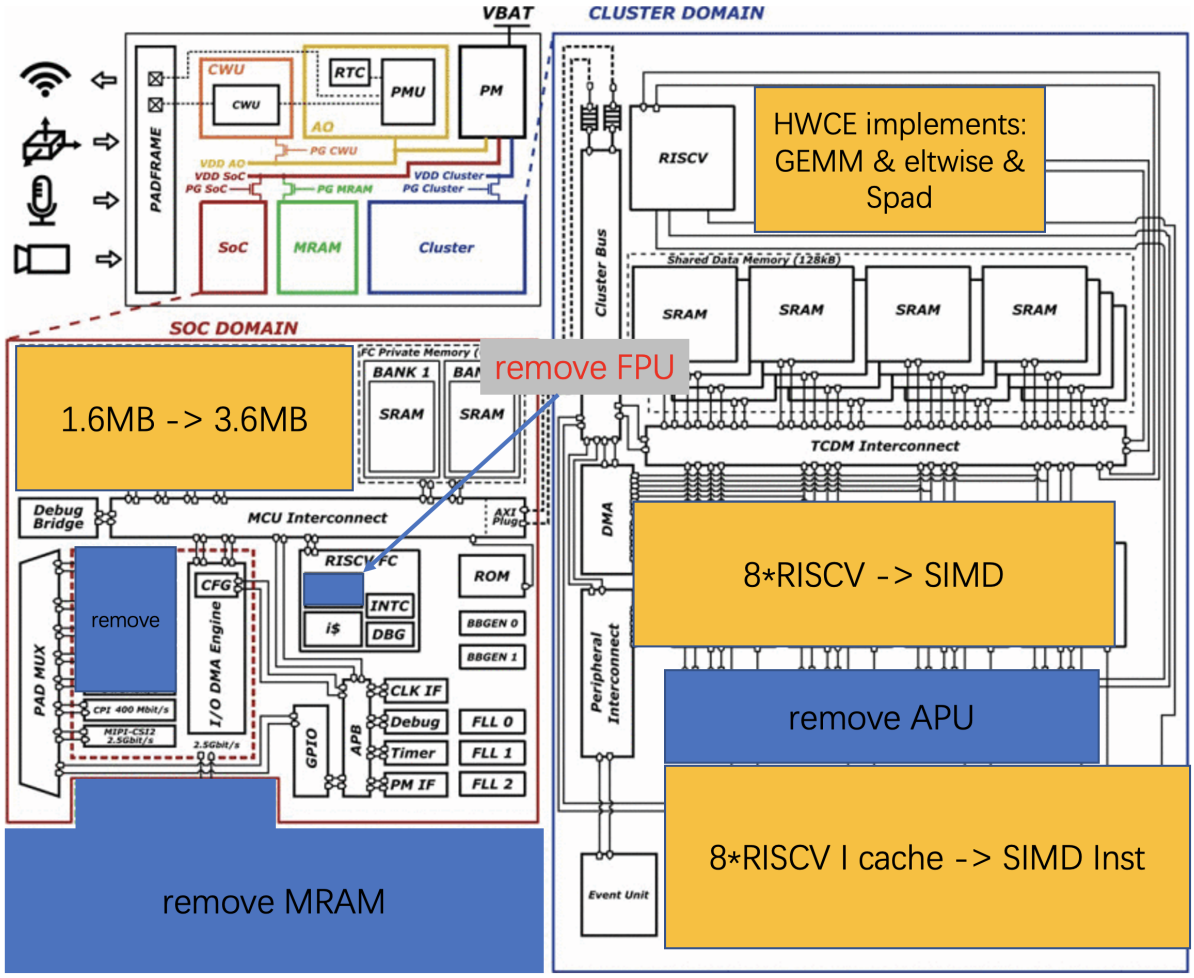

Fig. B3 Our design to refine the original SoC Layout.