Queue Computation Mechanism For Parallel Execution in Parallel Queue Processor

AKANDA MD. MUSFIQUZZAMAN ,† BEN. A. ABDERAZEK ,†
SOICHI SHIGETA ,† TSUTOMU YOSHINAGA † and MASAHIRO SOWA †

In this paper, we describe the architecture of Queue Computation Unit (QCU) that is implemented in a Produced order Parallel Queue Processor. The QCU effectively calculates the queue head and tail values for each instruction. The QCU has two major hardware parts: (1) Queue Computation Circuit (QCC), and (2) Queue Buffer (QB). We first give a brief introduction showing how the Queue Computation Unit is interfaced to the PQP processor. Then, we present the QCU mechanism architecture.

1. Introduction

Parallel Queue Processor (PQPpfB) architecture is a novel processor architecture that stores data in a First-In-First-Out (FIFO) scheme and exploits parallelism dynamically[1)2)]. The PQPpfB stores, data in produced order scheme. The architecture has six pipeline stages: (1) Fetch unit, (2) Decode unit,(3) Queue computation unit, (4) Barrier queue & control unit, (5) Issue unit & (6) Execution unit. It can issue up to four instructions in parallel and per cycle. As a result, the PQPpfB is expected to have much lower hardware complexity than conventional architecture[3)4)]. The QCU, described in this paper, is one of several hardware units that forms the hardware of the PQPpfB processor. It calculates the Queue head and Tail values for each instruction. The algorithm which was used to calculate the Queue Head and Tail value is shown in Figure 1.

2. QCU in PQPpfB Processor

The QCU unit consists of two major components: (1) Queue head (QH) and (2) Queue tail (QT). The QH points to the head of the operand Queue (OPQ) and the QT tail indicates the end point of the OPQ. The QCU is responsible for calculating QH and QT values for each instruction. For each instruction, the QCU also calculates the live queue head (LQH) value which indicates the starting point from where the queue is alive. The calculated value of LQH, QH and QT will be used later by in the pipeline by the Issue Unit (IU).

† Graduate School of Information Systems, The University of Electro-Communications, Tokyo, Japan.

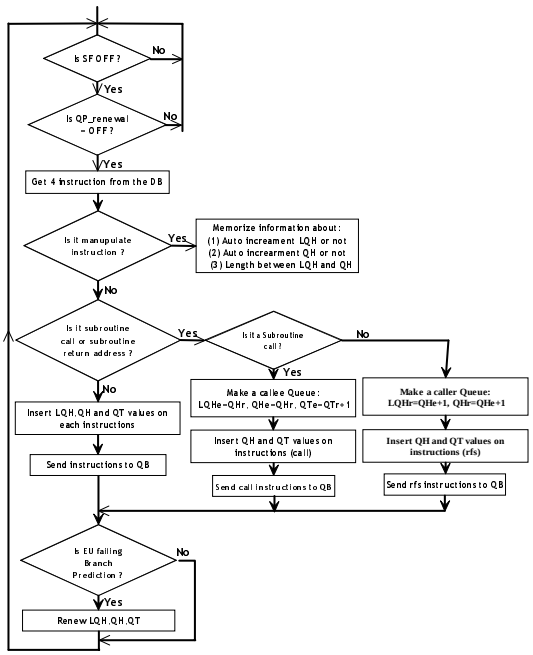

Fig. 1 Queue Computation Algorithm

3. Queue Computing Mechanism

As we mentioned, the QCU has two parts: (1) Queue Computation Circuit (QCC) & (2) Queue Buffer (QB). The QCU, which takes its input signals from the decode buffer has three 8-bit internal registers LQH, QH & QT. These three registers are used by the QCU for parallel calculating. There are four QCC units that are connected by wires. The inputs of each QCC are: (a) opcode, (b) operand, (c) delta_LQH, (d) delta_QH, (e) delta_QT and (f) flags. These are also 8-bit input. The Flags indicate the type of instruction. Although flags are 8-bit input,

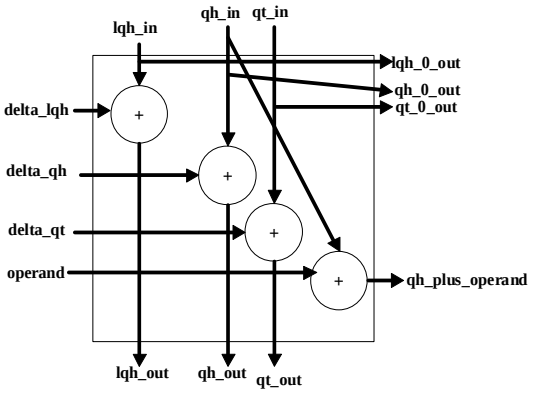

Fig. 2 Block Diagram of QCC

we are using only 1-bit of this 8-bit. When Flag = 0 the instruction type is normal instruction and if Flag = 1, the instruction type is immediate instruction. Here the word " delta" indicates the data fetched from the decode buffer of Decode Unit. The QCU also take three 8-bit inputs from the internal registers (LQH_in, QH_in and QT_in). The outputs (8-bit) are LQH_out, QH_out, qt_out. Figure 2 shows and internal representation diagram of a single QCC unit.

The calculation of QH, QT and LQH values is performed by the following formulas:
LQH_out = LQH_in + delta_LQH
QH_out = QH_in + delta_QH
QT_out = QT_in + delta_QT
QH_plus_operand = QH_in + operand

If the instruction type is Immediate Instruction
QH_plus_operand = operand

In the QCC the opcodes are totally unchanged. It just pass through the opcode to the QB without changing as :
opecode_out = opcode_in
The QB is also a buffer that can contains 8-bit 24 words. Actually, it is an output buffer of the QCU. It contains the calculated values of four instructions at a time. Figure 3 shows the whole internal works of QCU. Here, q_renew signals comes from the execution unit which indicate to renew the values of LQH, QH and qt.

4. Hardware Implementation Results

The QCU is an integral part of the PQPpfB processor. It was implemented and described in Verilog-HDL. It was correctly simulated as a single module and currently it is being verified, within the whole PQPpfB processor, with Alt era APEX20kE400 FPGA.

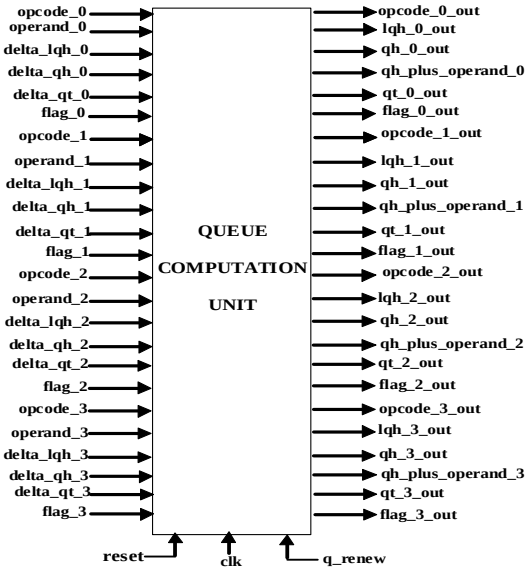

Fig. 3 QCU I/O interface

5. Conclusion

In this article we explained the queue computation mechanism that enables the parallel instruction execution in Parallel Queue Processor. For hardware implementation the HDL code is written in the verilog'2001 platform. Our future work is to optimise the QCU hardware and evaluate it within the PQPpfB processor.

References

1) M. Sowa, B. A. Abderazek, S. Shigeta, K. Nikolova, and T. Yoshinaga: *Proposal and Design of a Parallel Queue Processor Architecture (PQP)* , 14th IASTED Int. Conf. on Parallel and Distributed Computing and System, Cambridge, USA, (pp. 554-560, Nov. 2002),

2) B. A. Abderazek, S. Shigeta, T. Yoshinaga and M. Sowa: *Architectural Issues in the Design of a High Performance Parallel Queue Processor*, 4th Bilateral Symposium on Science & Technology, April 2003,

3) B. A. Abderazek, Soichi Shigeta, K. Nikolova, T. Yoshinaga, Masahiro Sowa: *Parallel Queue Processor Design*, IEICE CPSY2002-60, pp. 55-60, Nov., 2002,

4) Sowa Lab: *PQPpfB Project:* www.sowa.is.uec.ac.jp/sowalab/fpga/public