Custom VLSI ASIC for Automotive Applications with Recurrent Networks

R. Tawel and N. Aranki
Jet Propulsion Laboratory
Pasadena, CA **911098099**

G. V. Puskorius, K. A. Marko, L. A. Feldkamp,
J. V. James, G. Jesion, and T. M. Feldkamp
Ford Research Laboratory
Dearborn, MI 48121-2053

Abstract

Demands on the performance of vehicle control and diagnostic systems are steadily increasing as a consequence of stiff global competition and government mandates. Neural networks provide a means of creating control and diagnostic strategies that will help in meeting these demands efficiently and robustly. This paper describes a VLSI design that permits such networks to be executed in real time as well as the application, misfire detection, that served as a focus for the collaborative effort.

1 Introduction

The control system of a modern automobile involves several interacting subsystems, almost any one of which provides interesting theoretical and engineering challenges. Further, increasingly stringent emissions regulations require that any malfunctioning component or system with the potential to undermine the emissions control system be detected and identified. Neural networks have the potential for major impact in this work. Benefits may be anticipated in design time or performance of a control (Puskorius and Feldkamp, 1996) or diagnostic strategy (Marko et al., 1996). We have shown that both of these applications are suited to the use of recurrent multi-layer perceptrons, the architecture of which may be regarded as the joint generalization of a feedforward multi-layer perceptron and a one-layer fully time-lagged recurrent network. A potential barrier to the use of such networks, however, arises from the considerable burden from other functions already carried by existing powertrain processors. This prompted an effort to develop a VLSI design that would facilitate the implementation of recurrent networks in high-volume products.

2 Neuroprocessor Chip

The design constraints for this project called for the development of an inexpensive, fully autonomous, and commercially viable electronic chip. This single chip implement ation was required to be (1) ext remel y compact in size (mass market potential) (2) flexible (several neural based applications would share the hardware and sequentially execute on it), and (3) accurate (no miscalls due to limited hardware resolution). Observing that combustion events occur, even at maximum engine speed, on a millisecond time scale, a novel and extremely compact and powerful layer-multiplexed bit-serial neuromorphic architecture was developed and exploited for the VLSI CMOS implementation.

2.1 Architect ure

The required computations can be summarized as a series of parallel multiply and accumulate (MAC) operations intcrspersed by an occasional nonlinear operation. We exploited five basic stategies to achieve our desired goals: (1) parallel intra-layer topology; (2) single-instruction-multipk-data (SIMD) architecture; (3) bit-serial fixed-point computational techniques; (4) inter-layer multiplexing of neuron resources; and (5) nonlinearities handled by look-up-tables.

The resulting architecture is shown schematically in Figure 1 and consists of (1) a global controller; (2) a pool of 16 bit-serial neurons; (3) a ROM based bipolar sigmoid activation look-up-table; (4) neuron state registers; and (5) a synaptic weight RAM. In this design, both inputs to the network as well as neuron outputs

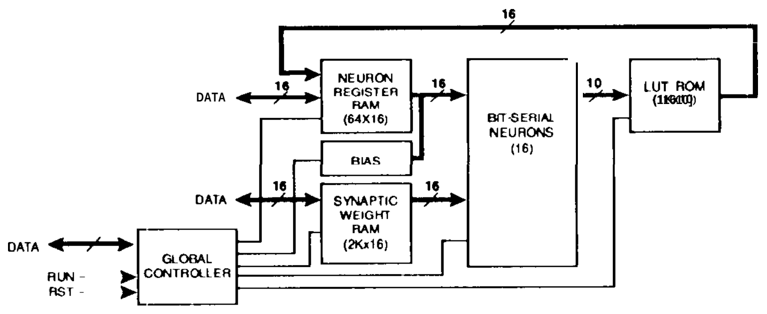

Figure 1: Schematic representation of forward propagation module.

are stored in the neuron state RAM. When triggered by the global controller, each of the 16 neurons performs the multiply and accumulate (MAC) operation. They receive in a bit serial fashion as input the synaptic weights (from the synaptic weight RAM) and activations from either (a) input nodes or (b) outputs from other neurons and output the accumulated sum of partial products onto a tri-stated bus which is commonly shared by all 16 neurons. Because of the computat ional nature of neural networks where information is sequentially computed a layer at a time only enough neurons are physically implemented in silicon as exist on the layer with the largest number of neurons for all applications of interest. As such, a candidate pool of 16 silicon neurons was chosen. This iutra-layer pool of neurons is organized in a SIMD configuration. Single-instruction (S1) means that all active neurons in the pool execute the same instruction at the same time. Multiple-data

(MD) means that each active neuron acts on its own slice of data, independently of all other processors. Thus the chip performs fully parallel computations under the supervision of the global controller.

A significant reduction in silicon real-estate was achieved by performing inter-layer multiplexing of the 16 neuron pool, i.e., the hardware used in calculating the activations of neurons in onc layer is reused for the calculation of neurons in another layer, since neurocomputations arc performed a layer at a time. We also used bit-serial algorithms extensively for arithmetic operations because their canonical nature and minimal interconnection requirements make thcm particularly suitable for efficient VLSI implementation.

2.2 Controller

At the heart of the neuroprocessor architecture is the global controller. The controller contains the logic to enable the neurochip to execute its task. This task is to load an architecture from RAM, and once triggered, to generate all necessary control signals in addition to orchestrate data movement on-chip and off-chip. When there arc no computations being pm-formed, the global controller remains in the idle state, signaling its availability by having the active low BUSY flag set high. When a LOAD command is issued, the controller reads from RAM a neural network topology and goes into an idle state. When the RUN command is subsequently issued, the global controller is in charge of providing control signals to the RAM, ROM, and the 16 on-chip neurons, in order to proceed with the desired computation. Input activations arc read out of the 64×16 Ncurou State RAM, synaptic weights are read out of the 2K x 16 Synaptic Weight RAM, and both arc propagated to the bank of 16 neurons. In this way, the global controller keeps track of both intra-layer operations as well as inter-layer operations. Upon completion of a forward pass through the network architecture, the global controller asserts the BUSY flag and returns to the idle state.

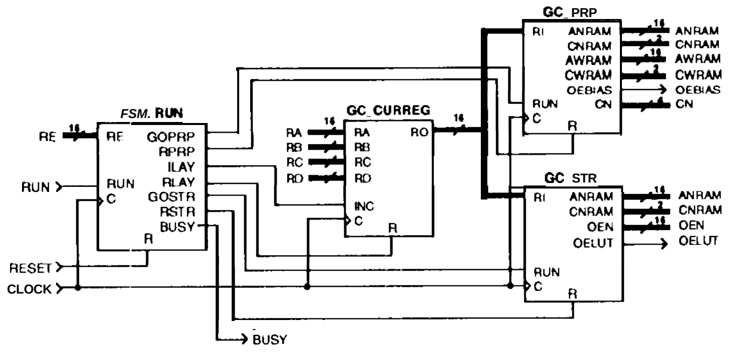

Figure 2: Run-time forward propagation controller.

2.3 Neurons

Fixed-point bit-serial algorithm for operations such as addition and mult iplication arc uniquely suitable for efficient VLSI implementations because of their highly compact representations. For example, the size of an n x n bit multiplier scales quadratically ($O(n^2)$) for a bit-parallel implementation and linearly ($O(n)$) for a bit-serial one. ~lit-serial techniques were therefore used. A schematic representation of a bit-serial neuron is shown in Figure 3.

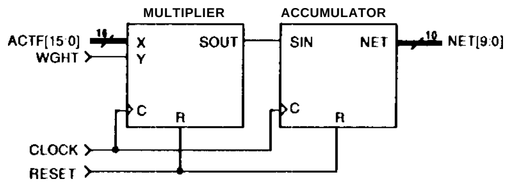

Figure 3: Ilit-serial neuron.

Precision constraints for the misfire problem called for the usc of a 16×16 bit fixed-point multiplier. In operation, the multiplier accepts as input either an input stimulus to the neural network or an activation output from a neuron on a previous layer. It multiplies this quantity by the corresponding synaptic weight. The input stimulus (or activation output) is presented to the multiplier in a bit-parallel fashion, while 'the synaptic weights arc-presented in a bit-serial fashion The serial output of the multiplier feeds directly into an accumulator.

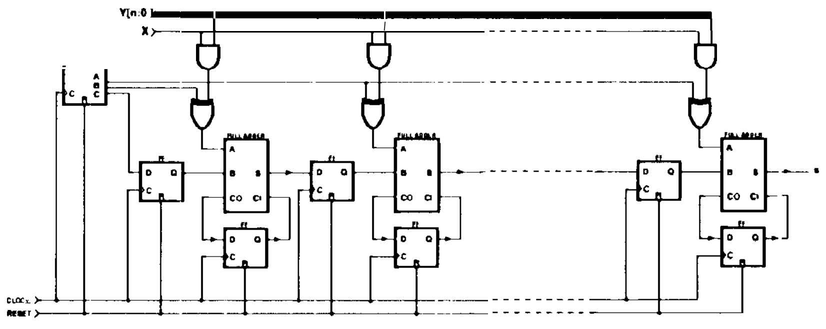

Figure 4: Hit-serial multiplier of length n

The multiplier shown in Figure 4 is a modified and improved version of a previousl y reported serial multiplier. Any size multiplier can bc formed by cascading the basic multiplier ccll. The bit-wise multiplication of the multiplier and multiplicand is performed by the AND gates. At each clock cycle, the bank of ANI) gates compute the partial product terms of the multiplier Y[15:0] and the serial multiplicand X (t). Two's complement multiplication is achieved by using XOR gates on the outputs of the AND gates. By controlling onc of the inputs of the XOR gate, the finite state machine FSM can form the two's complement of selected terms based on its control ffow. In general, for an n x n multiplier (resulting in a $2n$ bit product), the multiplier can bc formed by using $2n$ basic cells and will perform the multiplication in $2n$ + 2 clock cycles. Successive operations can bc pipelined and the latency of the LSB of the product is n + 2 clock cycles.

The accumulator, shown in Figure 5, is also a bit-serial design. It is extremely compact as it consists of a single bit-serial adder linked to a chain of data registers. The length of the accumulator chain is governed by the multiplication length. The multiplier takes $2n+2$ clock cycles to perform a complete n x n multiplication. At each clock cycle, the accumulator sums the bit from the input data stream with both the current contents of the data register on the circular chain as well as any carry bits that might have been generated from the addition in the previous clock

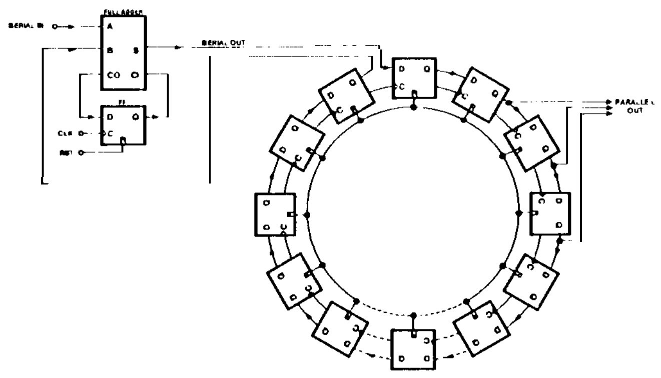

Figure 5: Ilit-serial accumulator of lcngth n.

cycle. This value is subscqucntly storcd onto the chain on thc next clock cyclc. This crcatcs a circulating chain of data bits in thc accumulator with pcriod $2n+2$.

The ncuroproccssor design was implemented using H P's 0.5 μm CMOS dcsign rules. Thc first generation chip mcasurcd $8mm^2$ in sizc. The current design operates at a conscrvativc 20 MHz clock speed. A neural application) can bc loaded into the **hardware in** undcr 1 μs. Bccausc of the S1 MD architccturc, it takcs 1.6 μs to simultaneousl y perform 16 multiply and accumulate opcrations. This translates into an cffcctive computational throughput of 0.1 μs pcr MAC operation. Thc next gcneration proccssor will operate at 50 MHz.

3 The Misfire Diagnostic Problem

Bccausc engine misfire can cause a significant, incrcase in tailpipe cmissions and can damage the catalytic convcrtcr, it is a required diagnostic. Misfire dctcction must bc performed bctwccn engine cylinder firings, which can occur at rates as high as 30,000 events pcr minute, so that approximately onc billion cvcnts must bc classified over the life of each vchiclc. While thcrc arc many ways of detecting enginc misfire, all currently practical methods rely on obscrving enginc crankshaft rotational dynamics with a position sensor locatcd at onc cnd of the shaft,. Briefly stated, onc looks for a crankshaft acceleration deficit following a cylinder firing and attempts to dctcrminc whether such a deficit is attributable to a lack of power provided on the most rccent firing stroke. The mcthod is complicated by several factors: 1) the crankshaft dynamics arc influenced by unregulated inputs from the driver and disturbanccs introduced through the drivcshaft from road irregularities; 2) thc dynamics arc obscured by mcasurcmcnt noisc; 3) the crankshaft is not infinitely stiff and exhibits complex dynamics which mask the signature of thc misfire event and which arc influenced by the event itself. In effect, wc arc observing the torsional oscillations of a nonlincar oscillator with driving forces applied at several locations along its maiu axis. While it is straightforward to write down dynamical equations that approximate thc crankshaft rotational dynamics as a function of the combustion pressures applied to the piston faces, it is difficult, to solvc those equations and even more difficult to solvc thc inverse infcrencc problcm associated with misfire diagnostics. Nonctheless, it was thc expect ation of a discoverable dynamic rclationship bctwccn the observed accelerations and thc driving forces in this systcm, coupled with thc abscncc of a satisfactory altcrnat ivc approach, that motivated our exploration of rccurrent networks as a solution to thc problcm.