

A RECONFIGURABLE VITERBI DECODER FOR A COMMUNICATION PLATFORM

Imran Ahmed^{*†}

^{*} School of Electronics and Engineering,
University of Edinburgh, King's Buildings
Mayfield Road, Edinburgh, EH9 3JL, UK
email: imran.ahmed@sl-i-institute.ac.uk

Tughrul Arslan^{*†}

[†] Institute for System Level Integration, The
Alba Campus, The Alba Centre, Livingston,
Scotland, EH54 7EG, UK
email: T.Arslan@ed.ac.uk

ABSTRACT

A new large constraint length, soft decision viterbi decoder fabric is presented for deployment using platform based system on chip methodologies. The decoder can be reconfigured for standards such as CDMA2000, WCDMA (UMTS), ADSL, IEEE 802.11, and GSM. Maximum resource allocation and performance is achieved by reusing components within turbo decoder base array. This cross platform viterbi decoder is reconfigurable between different trellis types, constraint lengths and rates making it ideal for a unified multi-standard telecommunication platform. In addition, the authors also propose a novel technique for dynamic reconfiguration in order to achieve faster context switching between different mappings. The reconfigurable fabric is implemented as a subset of turbo decoder array on a 180 nm UMC process technology.

1. INTRODUCTION

A Viterbi decoder is an important subsystem of any wireless communication receiver. It is based on the technique of a-posteriori estimation of the state sequence for channels with memory less noise. The main aim of this reported research is to introduce a viterbi decoder architecture that can be reconfigured to decode a range of convolutionally coded data. The architecture can support up to 256 states, trellises with different generator polynomials and rates. The design is mapped on to the turbo decoder array and these blocks together provide forward error correction for a reconfigurable communication platform. The flexibility of the reported design is within restricted domains as compared to fine grained gate arrays. This flexibility trade off provides improved performance in terms of area and power.

Previous work on unified Viterbi-Turbo decoding has normally been limited to 3G codes [1]; our work provides an open trellis structure for both viterbi and turbo decoding for multiple standards. A systematic approach adopted for partitioning, scheduling and mapping onto an area-efficient architecture is presented. The proposed architecture reuses the available 8 Add-Compare-Select (ACS) blocks in turbo decoding array. The concept of reusing these blocks results

in a higher speed architecture than traditionally used state serial architectures [2-3]. In comparison to the fully parallel architectures [4] an intermediate solution is presented where better area efficiency can be achieved by Processing N states (up to 256) by P ACS (P=8) [4] in contrast to assigning one ACS for each state. The state machine control makes it possible to use an array as a co-processor. We have also shown a reconfigurable trace back approach for survivor memory management.

2. VLSI DESIGN

The aim of this paper is to describe the viterbi VLSI design and mapping decisions on turbo decoder array. Therefore the turbo decoding codec and viterbi decoding algorithm have not been discussed in great details. The overall block diagram of viterbi decoder is shown.

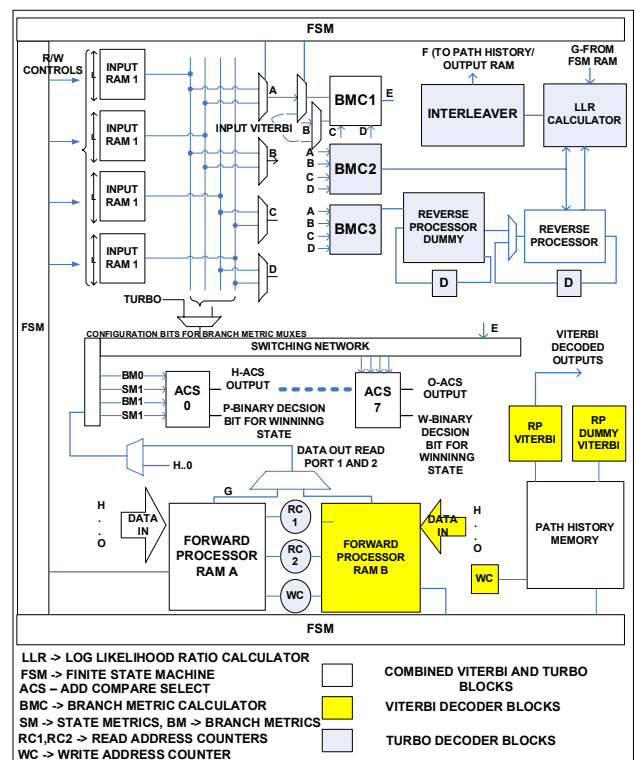


Figure 1. Block diagram.

2.1. ACS Block.

For coherence of representation the viterbi algorithm (VA) is summarized with a 2 state trellis diagram. For each stage 'L' of trellis, branch metrics (BM) are computed using the soft input symbols. State metrics (SMs) for stage L+1 are updated using the SMs for stage 'L' of trellis as shown in figure 2.

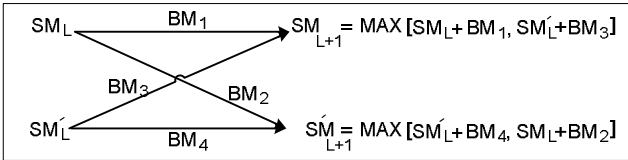


Figure 2. A simple butterfly operation for SM calculation.

The BMs are computed by calculating the Euclidian distance of the soft input metrics. The input metrics are represented in 4Q2 signed two's complement format similar to the turbo decoding array. The BM block for turbo decoder and viterbi decoder arrays is therefore similar.

The error probability of convolution codes decreases exponentially with the constraint length [5] and therefore standards like 3GPP [7] use large trellises. For brevity we will explain the discussion with 3GPP example with 256 states (constraint length=9).

For this constraint length (K=9), the 256 (2^{K-1}) states are represented by a smaller de Bruijn graph of 2^M states [5]. We have 8 ACS blocks available in the turbo decoder array [6] and therefore M=3, processing 8 states/clock cycle for Viterbi. This is shown in the figure 3:

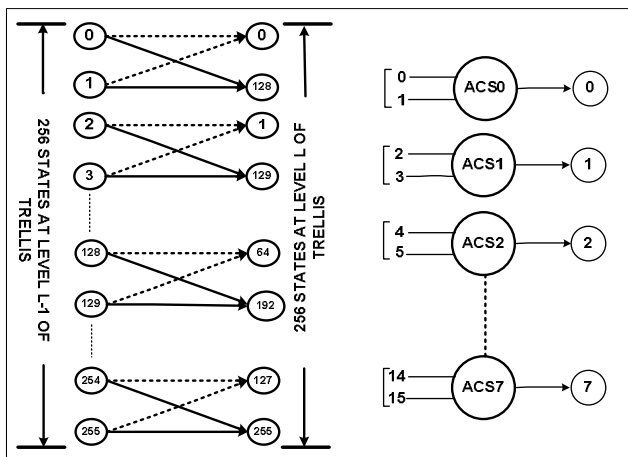


Figure 3. 256 states 3GPP trellis for generator polynomial 753,561

As shown in Fig 3 each ACS unit requires two states at trellis stage 'L-1' to calculate the winning state.

2.2. Path Metrics (PM) Memory.

There are 256 winning states at each stage 'L' of the trellis which are saved in SRAMs to be used at stage 'L+1'. Each stage (L) of trellis requires $2^{L-1}/2^M$ clock cycles to calculate all winning states. Dual read ports have been used to access two path metrics required for the computation of the next path metric. To avoid the read-write conflict two alternate RAM banks have been used for read and write operation. The update sequence is shown by the following pattern.

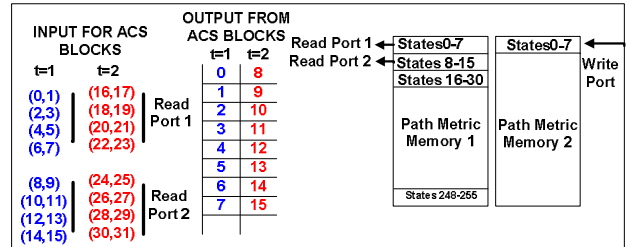


Figure 4. PM Memory read and write operations

At time t=1 'Read Port 1' read states 0-7 (for ACS 0-3) and 'Read Port 2' read states 8-15 (for ACS 4-7) from PM Memory 1. The ACS blocks update the path metrics and these are saved on PM Memory 2 as shown in blue. The process continues for trellis stage 'L' and the PM RAM2 is completely updated in 32 clock cycles. These values are then read in trellis stage 'L+1' and now PM Memory 1 will be used for writing the updated metrics. For lower constraint lengths (for e.g. GSM, K=5, 16 states) PM RAMs are updated in just two clock cycles.

The input/output state connections for the ACS blocks are explained in figure 5. Total number of states = 2^{a+1} .

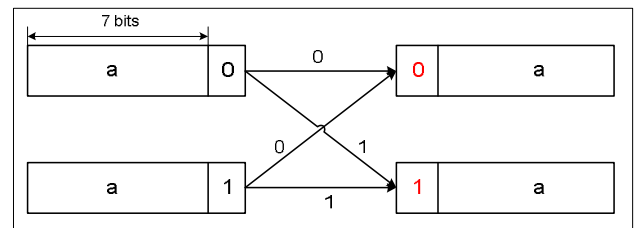


Figure 5. Next and Previous state calculation for all trellises

For example if the state at stage 'L' of trellis is 0_111111 (figure 6); it implies that the decoded bit (shown in red) is 0 and the two possible states connected to this winning state at stage 'L-1' of trellis are 111111_0 and 111111_1. The previous winning state decision is provided by the path history memory.

Path Metrics memory is mapped on to Forward State Metrics RAM [6] already available in turbo decoding array.

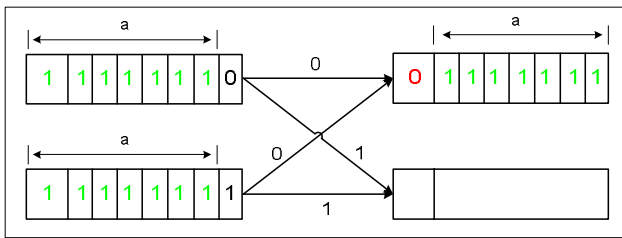


Figure 6. Example showing previous state calculation

2.3. Path History (PH) Memory.

Path History Memory is used in VA to find the survivor path. The contents of this memory are updated on each stage 'L' of the trellis which allows reconstructing the survivor path. Each ACS unit output one bit for the survivor state. 8 decision bits per clock cycle (given by 8 ACS units) are saved in the PH RAM. Total size of the PH RAM is given by:

Size of PH RAM = $(2^{L-1}/2^M) \times WL \times \text{Total windows}$ -----
 --- where window length is 6 times the constraint length.

For constraint length 9, size = $32 \times 54 \times 4$, 6912x8.

There is 8K (2Kx4) of output RAM available in turbo decoding array [6]. This is reused for PH Memory and is shown in the figure 7. Each block of 2K RAM is used to

store one window length (WL) of decision bits. The RAM is segmented by total states of trellis. For example, for 256 states there are 256 decision bits (one for each state). Therefore the RAM is segmented in to 32x8 segments. The used area of these four RAMS for each of the mapped standards is identical. Figure 6 shows mappings and segmentation of the PH memory for different standards. However the used area of each block of 2K RAM for a particular mapped standard will exactly be the same. This is shown by the table below

Standard	Constraint length	Memory utilization for each 2K RAM=WL x segment size.
W-CDMA(Japan) CDMA 2000 UMTS	9	$54 \times 32 = 1728 \times 8$ bits
GSM,PDC	5	$30 \times 2 = 60 \times 8$ bits
IS-95, IEEE 802.16	7	$42 \times 8 = 336 \times 8$ bits
IS-54	6	$36 \times 4 = 144 \times 8$ bits

Table1. Memory utilization for different standards.

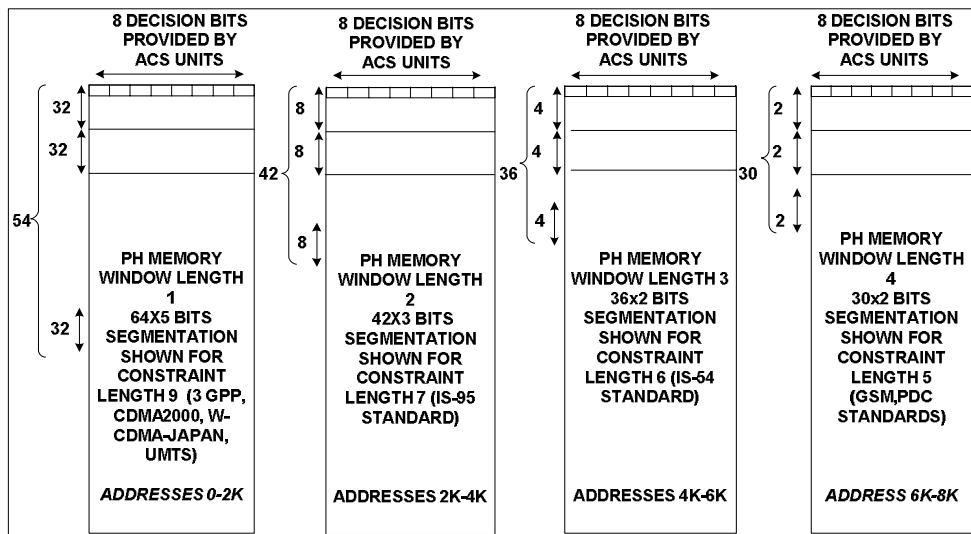


Figure 7. 2Kx4 identical path history RAMs, Segmentation and mappings shown for different standards.

PH memory is read by trace back processors (section 2.5) as they calculate the survivor path in the reverse traversing of the trellis. The write and read control is provided by the state machine. This is explained in the section 2.4. Address generation is made reconfigurable as explained below

2.3.1 Reconfigurable write address generation.

The write address generation is controlled by a 5-bit counter driving the 6-bit counter. The 6 bit counter will be

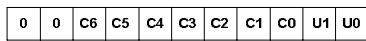
incremented once the 5 bit counter reaches the terminal (maximum) count. Both these counters have dynamic 'count_to' flags which provide the terminal count. As shown in figure 6 a technique of segment by segment address writing on PH memory has been used where each segment corresponds to one trellis stage 'L' (256 states for 3GPP). 5 bit counter controls the segment address. The segment address however is different for different standards and the maximum value is 5 bits for 3GPP. 6 bit up counter is concatenated with 4 zero bits on the MSB side and is then left shifted depending on the mapped

standard. The output of the arithmetic shifter drives the lower end of write address through tri state buffers. 5 bit up counter drives the other side of address generator through tri state buffers (B1-B4) as shown in the figure 7.

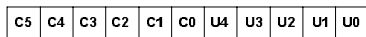
No	B1	B2	B3	B4	B5	B6	B7	B8	SH
1	On	On	On	On	Off	Off	Off	Off	4
2	Off	Off	Off	Off	On	On	On	On	0
3	Off	Off	On	On	On	On	Off	Off	2
4	Off	Off	Off	On	On	On	On	Off	1

Table 2. Configuration bits for tristate buffers

Table 2, defines configuration bits for the tri state buffers shown in figure 8. Rows in the table correspond to constraint lengths 9,5,7,6 respectively. The constraint lengths are for different standards as defined in table 1. Last column ‘SH’ in table 2 defines the no of left shifts for the arithmetic shifter. For example if 6 bit up counter gives $[C_5_C_4_C_3_C_2_C_1_C_0]$ and 10 bits arithmetic shifter input is $[0_0_0_0_C_5_C_4_C_3_C_2_C_1_C_0]$ then if SH is 1 (left shift by 1) the output from arithmetic shifter will be $[0_0_0_C_5_C_4_C_3_C_2_C_1_C_0_0]$. This output will drive the lower end of write address as shown in figure 7. Using table 1 and configuration switch values corresponding to SH 1 (line No 4 table 1), the write address will be



$[U4_U3_U2_U1_U0]$ are the five bit count values of upper counter (connected with B1-B4). This is the write address for IS54 standard, similarly for 3GPP the write address as shown by switches in line No 1 table 1, will be



Both these counters have count_to flags and for IS54 count_to flag of five bit counter will be set to 4, whereas count_to flag of 6 bit counter will be set to 36. For 3GPP the count_to flags of 5 bit and 6 bit counters will be 32 and 54 respectively.

2.4. State machine.

Viterbi state machine control is very similar to turbo decoding. The only difference between two is that the control is for read and write of Path history memory as opposed to input memories in turbo decoding. Both Turbo and Viterbi decoders use forward and reverse state metrics processing. There are windowed versions of the algorithm to improve the latency and in its simplistic form it uses two reverse processors (B1 and B2) in parallel with one forward processor (FP1).

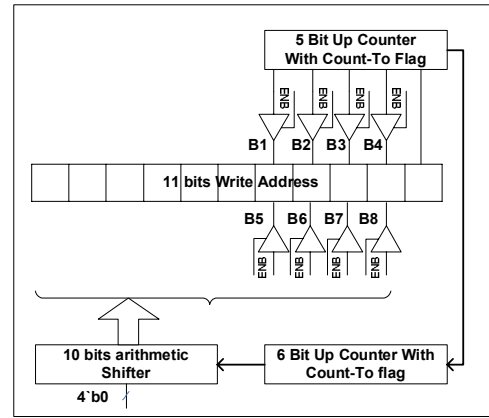


Figure 8. Reconfigurable write address generator

Reverse processor can start cold in any state (initializing each state as equi - probable), but after few iterations (equal to window length WL) the state metrics are as reliable as if the process had been started at the final node of trellis. Let B2 be the dummy reverse processor that starts from state 0 and after reverse traversing the trellis for a WL, provides the start state for the actual reverse processor B1.

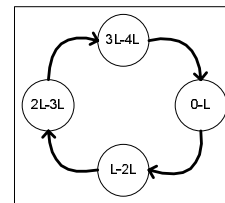


Figure 9. State machine

Figure 9 shows the four basic states of VA with start state as ‘0-L’. The detail scheduling diagram is presented in figure 10. Vertical axis preset time and horizontal axis present trellis length.

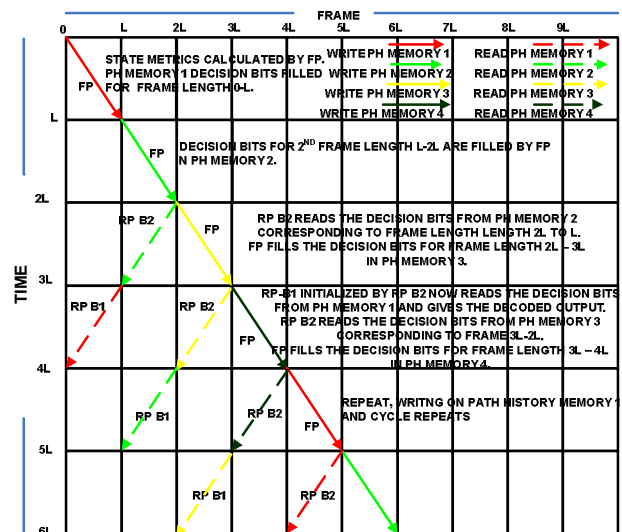


Figure 10. Scheduling diagram.

Table 3 is linked to Figure 9 for its explanation. Moreover a similar scheduling diagram for turbo decoding was presented in our earlier work [6]. Table 3 shows the working of FP, B1 and B2 as they write and read PH memories. After 4 WLs (0-L to 3L-4L) the cycle repeats. First decoded bits are output continuously after latency of 3 WLs from time 3L-4L.

Time	PH Mem1	PH Mem2	PH Mem3	PH Mem4
0-L	Write FP	Read B1	NO OP	Read B2
L-2L	Read B2	Write FP	Read B1	NO OP
2L-3L	NO OP	Read B2	Write FP	Read B1
3L-4L	Read B1	NO OP	Read B2	Write FP
4L-5L	Write FP	Read B1	NO OP	Read B2

Table 3. Read and writes on PH memories by FP,B1 and B2.

2.5. Reconfigurable Trace Back Processing.

As shown in the previous section there are two reverse processors B1 and B2 (dummy) working in parallel. To get the survivor path, either register-exchange or traceback structures can be used [8]. Since the trace-back is efficient for larger constraint lengths and low power applications, we chose trace back processing.

The previous trellis path stage S_{L-1} is given by the current path state S_L according to the following update.

$$S_{L-1} = [S_L \ll 1, D]$$

which corresponds to a left shift of the current state introducing the value of surviving bit D in the vacant position. This is also shown by D1-D7 in figure 11. Surviving bit is selected by multiplexer M1 from the data bus of PH Memories. The select control to this multiplexer is provided by D1, D2, D3 outputs. 6 bit Down Counter and arithmetic Shifter arrangement is also shown which works exactly the same as was explained in section 2.3.1. The only difference is that counter is initialized with the last address of PH Memory and counts down by 1. The decrement in count by one provides jump of one segment length. Reverse processor B1 and B2 share the same counter and shifter.

Reconfigurable trace back processing is explained with examples of 3GPP and GSM as was done in section 2.3.1.

	B1	B2	B3	B4	B5	B6	B7	B8
GSM	Off	off	off	off	on	on	on	on
3GPP	on	on	on	on	off	off	off	off

Table 4. Tri state buffer controls for reconfigurable trace back processing

Let $U_4_U_3_U_2_U_1$ are outputs of buffers B1-B2_B3_B4 and $C_5_C_4_C_3_C_2_C_1_C_0$ are the outputs of the 6 bit

down counter. Output of the arithmetic shifter with zero shift is $0_0_0_0_C_6_C_5_C_4_C_3_C_2_C_1_C_0$.

No	B9	B10	B11	sh	Output Shifter
GSM	off	off	off	0	$0_0_0_0_C_5_C_4$ $C_3_C_2_C_1_C_0$
3GPP	on	on	on	4	$C_5_C_4_C_3_C_2$ $C_1_C_0_0_0_0$

Table 5. Arithmetic shifter outputs and buffer controls

The contents of the read register for 3GPP using the controls in table 4 and table 5 will be

$C_5_C_4_C_3_C_2_C_1_C_0_U_4_U_3_U_2_U_1_U_0$

Similarly for GSM the read register contents will be

$0_0_0_0_C_5_C_4_C_3_C_2_C_1_C_0_U_0$

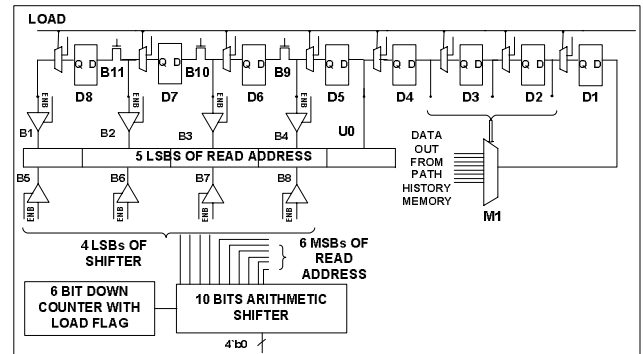


Figure 11. Traceback processing

2.6. Open Trellis and Dynamic Reconfiguration.

The input connections to all ACS units are made flexible by reconfigurable logic as shown in Figure-1. BM configurations are saved in Input RAMs 1- 4. Each RAM block is 32x8 bits, and has asynchronous read and synchronous write ports. The RAMs are filled in 32 clock cycles (for 3GPP [7] trellis). It is worth noting that during the write operation on the RAMs simultaneous read is also performed. This provides dynamic switch over for different trellis types. These configuration bits provide the appropriate BMs for the ACS units. ACS units also need the previous state metric values which are read from Forward Processor RAMs as explained earlier. After first segment write operation on the RAMs is completed and then only read operation is performed for the subsequent segments.

3. RESULTS

The design is synthesized using Synopsys Design Compiler for 0.18 microns CMOS UMC cell library and the chip layout is done on Silicon ensemble. Post layout power figures are taken from Synopsys Design Power by capturing the toggle activity of each node and then back annotating this in the circuit. Synopsys designware SRAMs were used for Forward Processor RAMs. Virtual Silicon 2K x 8 synchronous (separate read and write port) macro RAMs were used for Path history memory consuming 110 uW/MHz/Port.

The implementation uses clock gating to disable the unused blocks (shown by blue color in figure1). The overall area and the cost of reconfigurable components is shown in figure 11. Area utilization of reconfigurable switching is 97089.6 μm^2 which is only 3.4% of the overall area. The power consumption of this switching fabric is just 2.5% of the overall power consumption. Implementation of viterbi components on turbo decoder array increased the area of turbo decoder array by 20%. This overhead is much less than implementing viterbi and turbo decoder arrays separately. Active clock gating on disabled blocks results in no significant increase on power consumption of the overall array. The area and power figures are given in table 6. We have achieved similar power figures as compared to the unified VLSI design [1], however our design is made much more flexible than [1].

Area-Power Comparisons

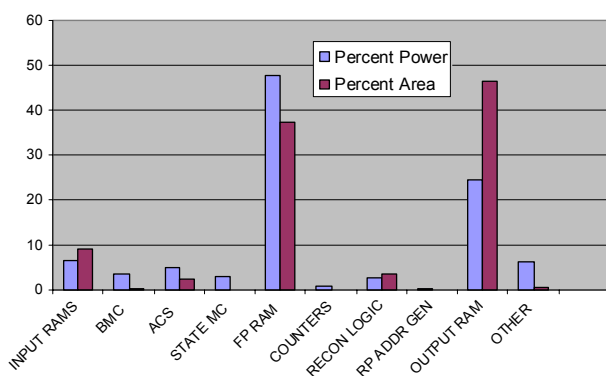


Figure 11. Area and Power results of individual components of the design

Technology	0.18 microns standard cell CMOS
Code rate (flexible)	1/2, 1/3, 1/4, 1/5
Constraint length (flexible)	Up to 9 (256 states)
Generator polynomial	Flexible
Survivor path length (flexible)	Maximum 54

Decision level	4 bit soft decision
ACS units	8
Power supply	1.8V
Operating frequency	20.0 MHz
Total Power in mW	69.961 mW
Total Area	2.82 mm^2

Table 1. Results.

4. CONCLUSION

A fully flexible viterbi decoder for reconfigurable platforms has been designed. The decoder consumes 69mw at 20MHz occupying 2.824 mm^2 area. The array can be mapped on to various communication standards and hence can be used as an IP in reconfigurable platforms. We have shown with results that the cost of reconfiguration in our chosen domain is negligible and a careful reconfigurable design can give results very close to the state of the art ASIC designs but with much increased flexibility.

5. REFERENCES

- [1] Mark A. Bickerstaff et. al., "A Unified Turbo/Viterbi Channel Decoder for 3GPP mobile wireless in 0.18- μm CMOS," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1555–1564, Nov. 2002.
- [2] Inyup Kang, et. al., "Low-Power Viterbi Decoder for CDMA Mobile Terminals," *IEEE J. Solid-State Circuits*, vol. 33, no. 3, pp. 473–482, Mar. 1998.
- [3] S. C. Glinski, et. al., "A processor for graph search algorithms" in Proc. ISSCC '87, New York, 1987, pp. 162–163.
- [4] C. Y. Chung and K. Yao, "Systolic array processing of the Viterbi algorithm," *IEEE Trans. Inform. Theory*, vol. 35, no. 1, pp. 76–86, Jan 1989.
- [5] C. B. Shung et. al., "Area-efficient architectures for the Viterbi algorithm – Part I: Theory," *IEEE Trans. Commun.*, vol. 41, pp 2907–2917, Sep 1993.
- [6] I. Ahmed, T. Arslan, "Improved Memory Strategy for LogMap turbo decoders," SOC Conference, 2005. Proceedings IEEE international pages 103–104, Sept 25–28 2005.
- [7] Third Generation Partnership Project, Technical Specification Group-Radio Access Network, Multiplexing and channel coding (FDD), 3G TS 25.212 v3.3.0.
- [8] C.M. Rader, "Memory management in a Viterbi decoder," *IEEE Trans. Commun.*, vol. COM-29, pp. 1399–1401, Sept. 1981