VA10V 1024 State 3GPP2 Viterbi Decoder

30 October 2013 (Version 1.02) Product Specification

VA10V Features

- 1024 state (constraint length 11) 3GPP2 compatible Viterbi decoder
- Up to 258 MHz internal clock
- Up to 14.4 Mbit/s
- Rate 2/3, 1/2, 1/3 or 1/4
- Optional or standard code polynomials
- 8–bit signed magnitude or two's complement input data
- Optional continuous, terminated (1 to 246 data bits) or tail–biting (16 to 256 data bits) decoding
- Estimated channel bit error outputs
- Optional serial (continuous only) or parallel data input
- Optional automatic coded symbol synchronisation for rate 1/2 QPSK and rate 1/2 to 1/4 BPSK
- LUTs: Virtex–4 8342, Virtex–5 7292, Virtex–6 7294, Spartan–6 7287, Artix–7 7306, Kintex–7 7605. 16 16Kb BlockRAMs.
- Asynchronous logic free design
- Free simulation software
- Available as EDIF core and VHDL simulation core for Xilinx Virtex–II, Spartan–3, Virtex–4, Virtex–5, Virtex–6, Spartan–6 and 7 Series FPGAs under SignOnce IP License. Actel, Altera and Lattice cores available on request.
- Available as VHDL core for ASICs

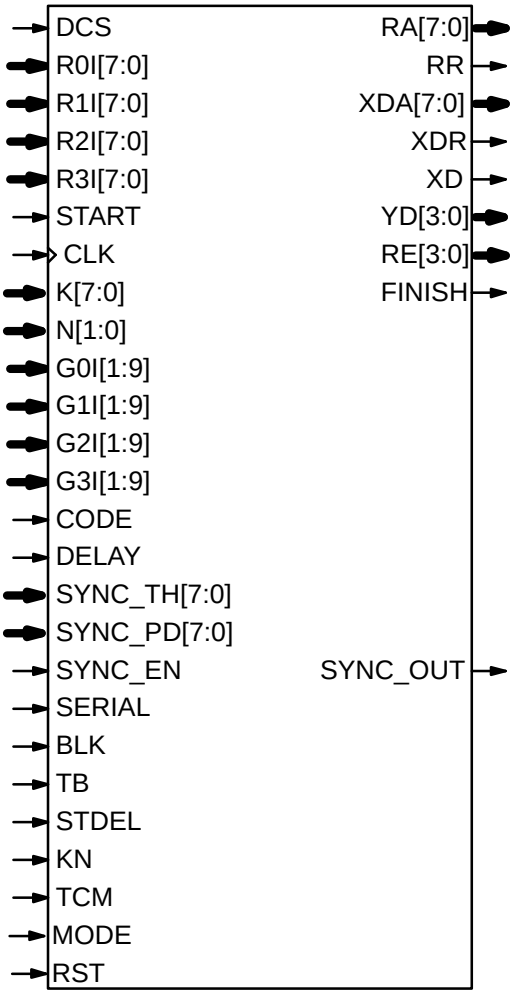

Figure 1: VA10V schematic symbol.

Introduction

The VA10V is a 1024 state error control decoder using the maximum likelihood Viterbi algorithm [1]. The decoder is designed to decode 3GPP2 Reverse Traffic Channel [2] encoded data as well as other custom coding solutions.

The decoder can decode continuously or with short terminated or tail–biting blocks. Internal depuncturing is available for rate 2/3 blocks. With continouous operation, automatic synchronisation is available for rate 1/2 to 1/4 BPSK serial and rate 1/2 QPSK. For best performance in fading channels, 8–bit sign–magnitude or two's complement inputs can be used.

To reduce complexity, the VA10V uses 64 12–bit add–compare–select (ACS) circuits in parallel 16 times to decode 1024 state convolutional codes. An internal 4Kx64 simple dual port synchronous RAM (implemented with 16 4Kx4 BlockRAMs) is used to perform the traceback. In synchronous operation, 18 clock cycles are required per decoded bit. Asynchronous operation requires 19 clock cycles.

Figure 1 shows the schematic symbol for the VA10V decoder. The EDIF core can be used with Xilinx Integrated Software Environment (ISE) software to implement the core in Xilinx FPGA's. Custom VHDL cores can be used in ASIC designs. Table 1 shows the performance achieved with various Xilinx parts (parallel input and no automatic synchronisation). T_{cp} is the minimum clock period over recommended operating condi-

tions. These performance figures may change due to device utilisation and configuration.

Table 1: Performance of Xilinx parts.*

Xilinx Part	T_{cp} (ns)	f_d (Mbit/s)
XC6SLX25–2	8.085	6.87
XC6SLX25–3	7.312	7.60
XC4VLX15–10	6.892	8.06
XC4VLX15–11	5.854	9.49
XC4VLX15–12	5.157	10.77
XC5VLX30–1	5.943	9.35
XC5VLX30–2	5.031	11.04
XC5VLX30–3	4.425	12.55
XC6VLX75T–1	5.279	10.52
XC6VLX75T–2	4.657	11.93
XC6VLX75T–3	4.233	13.12
XC7A100T–1	6.871	8.09
XC7A100T–2	5.745	9.67
XC7A100T–3	5.089	10.92
XC7K70T–1	5.021	11.06
XC7K70T–2	4.261	13.04
XC7K70T–3	3.871	14.35

*Synchronous operation

Signal Descriptions

BLK	Decoder Operation 0 = Continuous 1 = Block
CLK	System Clock
CODE	Code Select (see Table 2) 0 = 3GPP2 Polynomials 1 = Use G0I to G3I
DCS	Decoder Chip Select
DELAY	Decoder Delay Select 0 = 138 1 = 266
FINISH	Decoder Finish
G0I–G3I	Code Polynomial Input
K	Data Length
KN	Punctured Code Rate 0 = No Puncturing 1 = Rate 2/3
MODE	Maximum Rate Select 0 = Rates 1/2 to 1/3 1 = Rates 1/2 to 1/4
N	Convolutional Code Rate 2 = Rate 1/2 3 = Rate 1/3 0 = Rate 1/4
R0I–R3I	Received Data
RE	Estimated Symbol Error
RST	Synchronous Reset
SERIAL	0 = Parallel Input (R0I to R3I) 1 = Serial Input (R0I only)
START	Decoder Start
STDEL	Internal Start Delay (BLK = 1) 0 = One Clock Cycle Delay 1 = Two Clock Cycle Delay
SYNC_EN	Synchronisation Enable
SYNC_OUT	Synchronisation Output SIgnal
SYNC_PD	Synchronisation Period (1 to 255)
SYNC_TH	Synchronisation Threshold (1 to 255)
TB	Block Operation 0 = Terminated 1 = Tail Biting
TCM	Input Data Format 0 = Sign Magnitude 1 = Two's Complement
XD	Decoded Data Output
YD	Decoded Symbol Output

Table 2: Convolutional Codes.

N[1:0]	g0	g1	g2	g3
10	2327	3175	–	–
11	2373	2671	3165	–
00	2327	2353	2671	3175

Code Selection

Figure 2 gives a block diagram of a 1024 state (constraint length 11) non–systematic encoder. X is the data input and Y0 to Y3 are the coded outputs. GiIj = $g_i^j \in \{0,1\}, 0 \le i \le 3, 1 \le j \le 9$, correspond to the code polynomial coefficients which are used by the decoder.

The encoder polynomials are defined as

$$g_i(D) = 1 + g_i^1 D + g_i^2 D^2 + ... + g_i^9 D^9 + D^{10} \quad (1)$$

where D is the delay operator and + indicates modulo–2 (exclusive OR) addition. It is usual practice to express the coefficients in octal notation, e.g., $g_0 = 2327_8 = 10011010111_2 \equiv g_0(D) = 1 + D^3 + D^4 + D^6 + D^8 + D^9 + D^{10}$. This corresponds to G0I[1:9] = 001101011_2.

When CODE = 1, the code polynomials input to G0I[1:9] to G3I[1:9] are used. The input N[1:0] is also used to deselect the inputs for the various rates. That is, R2I[7:0] is internally grounded for rate 1/2 and R3I[7:0] is internally grounded for rates 1/2 and 1/3.

The 3GPP2 [2] convolutional code standards are selected by CODE = 0. The codes are given in Table 2 in octal notation. As the standard only

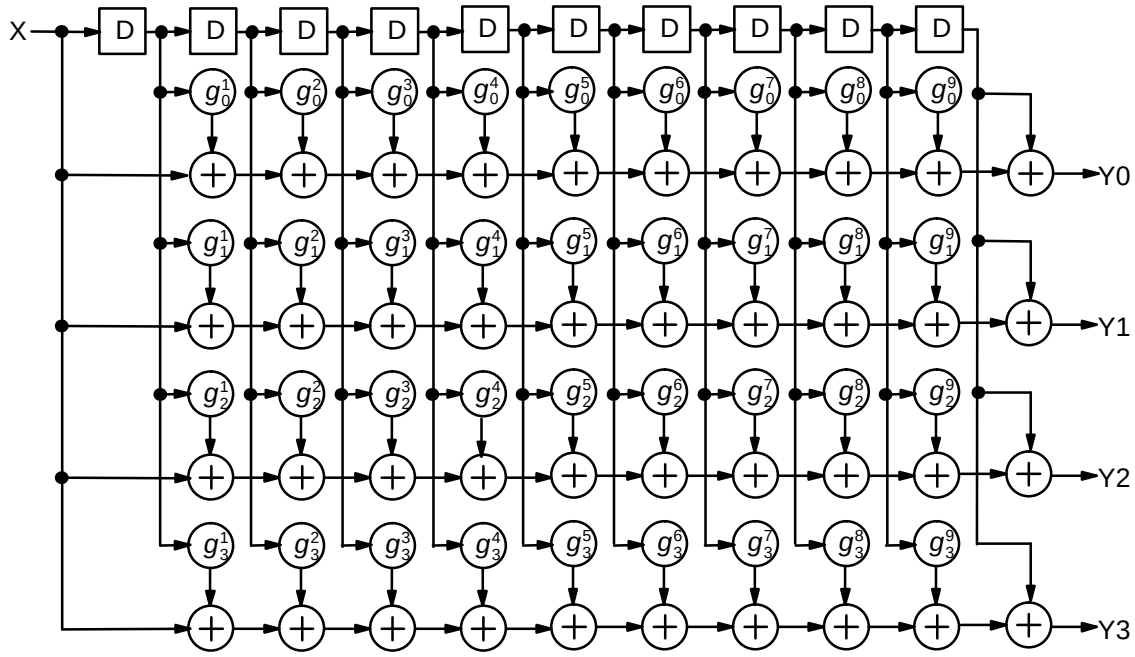

Figure 2: 1024 state non–systematic convolutional encoder.

gives code polynomials for rate 1/2 and 1/4, we chose the author's code from [3] for rate 1/3.

Note that the 3GPP2 standard specifies the code polynomials in a different octal format, for example 4656 for g_0, where the least significant bit zeros are ignored. We use the standard mathematical format where the most significant bit zeros are ignored. In both cases though, the same code polynomials are expressed.

Viterbi Decoder

The Viterbi decoder is designed to be very flexible and can be operated in either continuous or block mode.

Theory of Operation

The Viterbi decoding algorithm [1] finds the most likely transmitted sequence given the received noisy sequence.

For binary phase shift keying (BPSK) or quadrature phase shift keying (QPSK) modulation the received signal is described by

$$R_k^i = A((1 - 2y_k^i)/\sqrt{m} + n_k^i) \tag{2}$$

where A is the signal amplitude, $y_k^i \in \{0, 1\}$, $i = 0$ to 3 correspond to the coded bits, $m = 1$ for BPSK or $m = 2$ for QPSK, and n_k^i is a Gaussian distributed random variable with zero mean and normalised variance σ^2. Figure 3 shows the signal sets for BPSK and QPSK. We have

$$\sigma^2 = \left(2mR\frac{E_b}{N_0}\right)^{-1} \tag{3}$$

where E_b/N_0 is the energy per bit to single sided noise density ratio and $R = k/n$ is the code rate (k is the number of information bits and n is the number of coded bits).

Since a zero is transmitted as $+A/\sqrt{m}$ and a one is transmitted as $-A/\sqrt{m}$ the sign bit of a noiseless R_k^0 in two's complement notation is equal to d_k.

Due to quantisation and limiting effects the value of A should also be adjusted according to the received signal to noise ratio. A program called *cmap* for calculating the optimum values of A is included with the cores.

The value of A directly corresponds to the 8–bit signed magnitude inputs (described in more detail

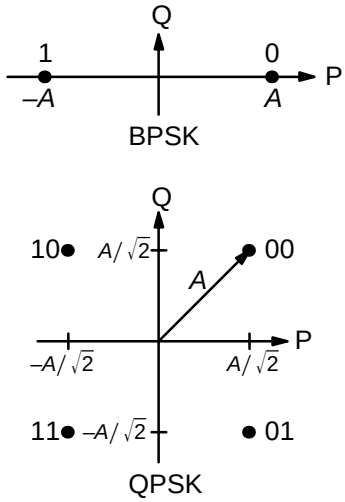

Figure 3: BPSK and QPSK signal sets.

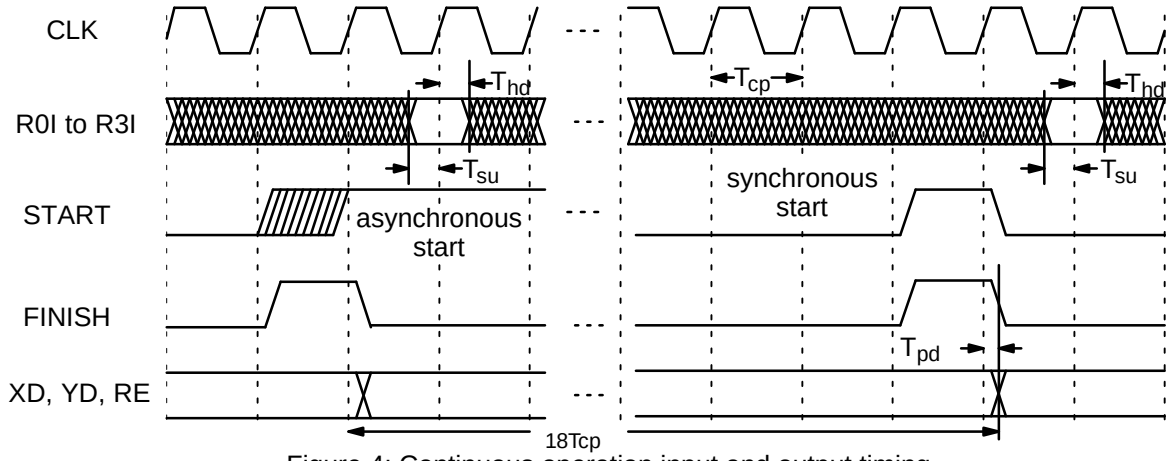

Figure 4: Continuous operation input and output timing.

later). The 8–bit inputs have 63 quantisation regions with a central dead zone. The quantisation regions are labelled from –31 to +31. For example, one could have A = 62.8. This value of A lies in quantisation region 63 (which has a range between 62.5 and 63.5).

Example 1: Rate 1/3 BPSK code operating at E_b/N_0 = 3 dB. From (3) we have σ^2 = 0.75178. Using cmap we have that A = 47.45.

Example 2: Rate 1/2 QPSK code operating at E_b/N_0 = 4 dB. From (3) we have σ^2 = 0.19905. Using cmap we have that A = 113.3. Note that the amplitude in each dimension is $A/\sqrt{2}$ = 80.12.

Decoder Operation

The VA10V uses a finite traceback memory and is thus able to continuously decode data. The traceback depth is determined by DELAY. Table 3 gives the minimum decoding depth, maximum decoding depth and decoder delay as a function of DELAY.

Table 3: Decoding depth and delay.

DELAY	Min Depth	Max Depth	Delay
0	113	120	138
1	225	240	266

The VA10V uses 64 12–bit ACS circuits in parallel. Thus, 16 clock cycles are required to perform 1024 ACS operations. An additional 2 clock cycle overhead is also required.

Continuous Operation

For continuous operation (BLK = 0), the decoder uses a rising edge detector circuit at the START input to start decoding the received data. If the high period of the START input is greater than the CLK period, the decoder will start decoding. To detect the next rising transition, the START input must be low for a least one CLK period.

This allows the decoder to be operated in synchronous or asynchronous operation. Synchronous operation requires 18 clock cycles per decoded bit. Asynchronous operation requires 19 clock cycles per decoded bit.

Figure 4 shows the relationship between the START input and R0I to R3I. In synchronous operation, these inputs must be valid from $2T_{cp}–T_{dsu}$ to $2T_{cp}+T_{dhd}$ after the rising edge of START (T_{cp}, T_{dsu}, and T_{dhd} are the decoder clock period, setup time, and hold time, respectively).

In asynchronous operation these signals must be valid from $T_{cp}–T_{dsu}$ to $2T_{cp}+T_{dhd}$ after the rising edge of START. Data must therefore change within one clock cycle after the rising edge of START.

The FINISH output goes high during the last clock cycle of the decoding operation. In continuous synchronous operation, the rising edges of START and FINISH should be coincident.

The decoded output XD, the re–encoded outputs YD[3:0] and estimated channel BER outputs RE[3:0] changes when FINISH goes low. RE[3:0] are obtained by exclusive ORing the appropriately delayed sign bit of the inputs with YD[3:0]. At low BER, these outputs can be used to give a good estimate of the channel BER.

Block Operation

For block operation (BLK = 1), the received data is stored in an external synchronous read input memory. The START signal goes high only once to start decoding. The outputs RR and RA are used to read the received data. For rate 1/2 operation, R2I and R3I are not used. For rate 1/3 operation R3I is not used. The output FINISH stays low until the last clock cycle of the last decoded bit.

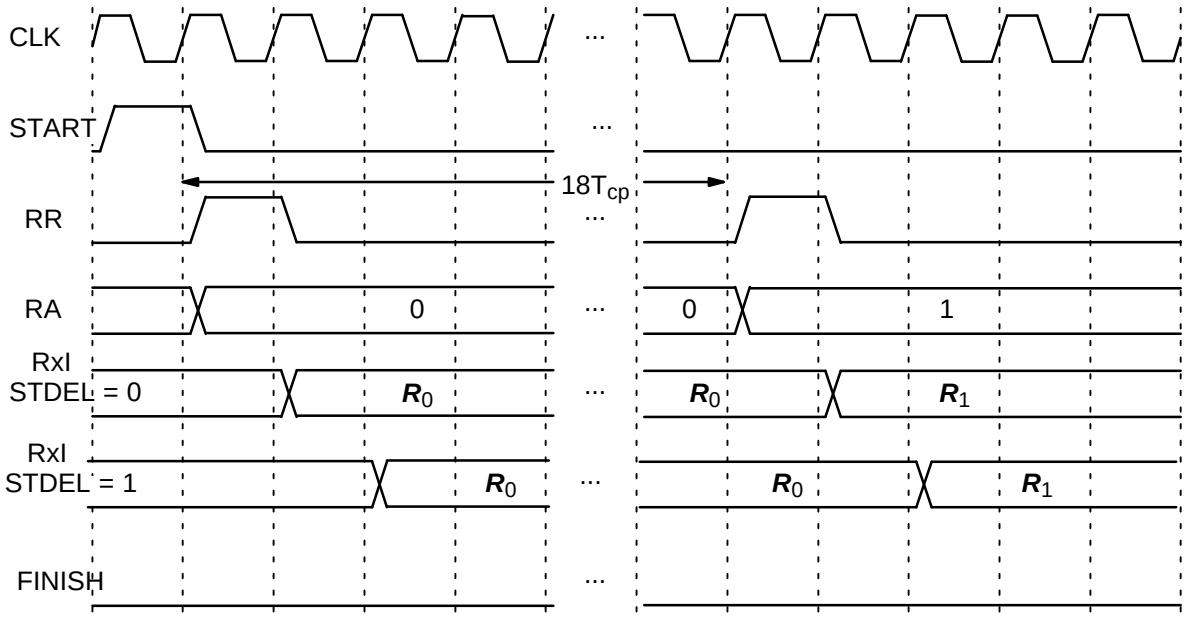

Figure 5: Terminated Input Timing.

The received data can be input either one clock cycle (STDEL = 0) or two clock cycles (STDEL = 1) after RR goes high.

For terminated codes (TB = 0), the input DELAY = 0 and 1 selects a delay of 138 and 266, respectively. For tail biting codes (TB = 1), the total delay is 266 and 522 for DELAY = 0 and 1, respectively. This assumes that the encoder start state is equal to the last $m = 10$ bits of the data block, where m is the encoder memory.

For TB = 0, the decoder first inputs the received data from address 0 to K–1, where K is the information data length which can vary from 1 to 246 bits. The tail is then input from address K to $K+m$–1. After a decoding delay, the decoded data is output to XD. XDR goes high for one clock cycle at the beginning of each decoded bit. XDA goes from address 0 to K–1 as the decoded data is output.

For TB = 1, the input sequence is more complicated. For DELAY = 0, the data is input for 128 symbols from address –128 mod K to K–1, for K symbols from address 0 to K–1, and then for 128 symbols from address 0 to 127 mod K. For DELAY = 1, the data is input for 256 symbols from address –256 mod K to K–1, for K symbols from address 0 to K–1, and then for 256 symbols from address 0 to 255 mod K. The decoder automatically calculates the correct address for RA, so the data only needs to be stored from address 0 to K–1.

Due to the modulus operation, the data lengths avalable for tail biting are restricted for use in the decoder. For DELAY = 0, data lengths from 16 to 256 bits can be used. For DELAY = 1, data lengths from 32 to 256 bits can be used.

Figures 5 and 6 shows the Viterbi decoder input timing for terminated and tail–biting codes, respectively. Either one or two clock cycle are used to start decoding (for terminated and tail biting codes, respectively), with each decoded bit taking 18 clock cycles. Figure 7 shows the Viterbi decoder output timing.

The decoding speed is given by

$$f_d = \frac{F_d}{18(1 + D/K) + S/K} \qquad (4)$$

where F_d is the internal clock speed, D is the total decoding delay in bits (shown in Table 4), and S = 2+TB+STDEL is the start delay (either 2, 3 or 4). For example, if TB= 0, STDEL=0, DELAY = 1, K = 192 and F_d = 200 MHz then D = 266, S = 2 and the decoding speed is 4.66 Mbit/s.

Table 4: Total decoding delay.

TB	DELAY	*D*
0	0	138
0	1	266
1	0	266
1	1	522

Input Data Format

The decoder internally uses 8–bit signed magnitude quantisation for R0I to R3I. External data can be input in either sign–magnitude (TCM = 0) or two's complement (TCM = 1) format. For two's

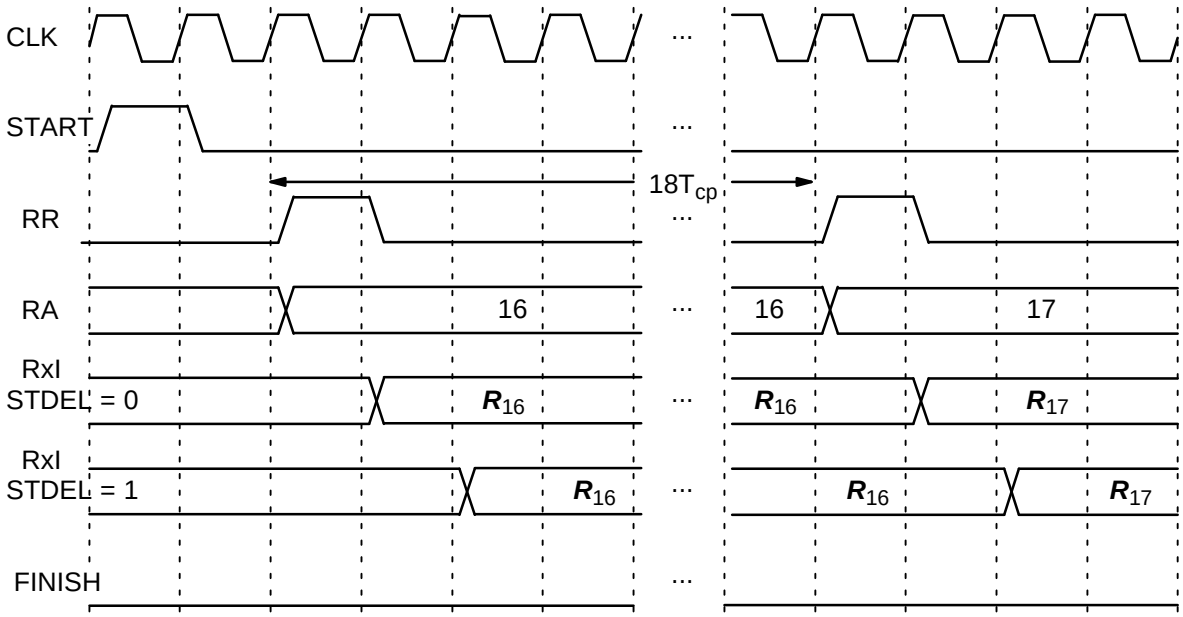

Figure 6: Tail Biting Input Timing (*K* = 48, DELAY = 0).

complement, inputs equal to –128 are internally limited to –127.

Tables 5 and 6 shows the 8–bit quantisation ranges for sign magnitude and two's complement inputs, respectively. Note that for sign magnitude, 0 and 128 indicate the central dead zone and have the same range. Note that most analog to digital to digital (A/D) converters do not have a central dead zone. For maximum performance, we recommend that 9–bit A/Ds are used with the output converted to 8–bit so that the appropriate ranges are obtained.

For input data quantised to less than 8–bits, the data should be mapped into the most significant bit positions of the input, the next bit equal to 1 and the remaining least significant bits tied low. For example, for 6–bit received data R0T[5:0], where R0T[5] is the sign bit, we have R0I[7:2] = R0T[5:0] and R0I[1:0] = 2 in decimal (10 in binary). For punctured input data, all bits must be zero, e.g., R1I[7:0] = 0.

Punctured Code Operation

Manual puncturing (KN = 0) can be performed by forcing R0I[7:0] to R3I[7:0] low. For example, rate 2/3 can be obtained by puncturing a rate 1/2 code with puncturing patterns of 11 for R0I and 10 for R1I. That is, R0I is not punctured, while R1I is forced low every other decoded bit.

Rate 2/3 internal depuncturing can be selected with BLK = 1, N = 2 and KN = 1. Data is input three symbols at a time every 36 clock cycles to R0I, R1I and R2I. The puncturing patterns for Y0 and Y1 are 1011 and 1110, respectively, as per the 3GPP2 standard.

Table 7 illustrates the order in which data is input to the decoder. The coded bit Y*i*(4*j*+*k*) corresponds to the coded bit Y*i* at coded symbol time

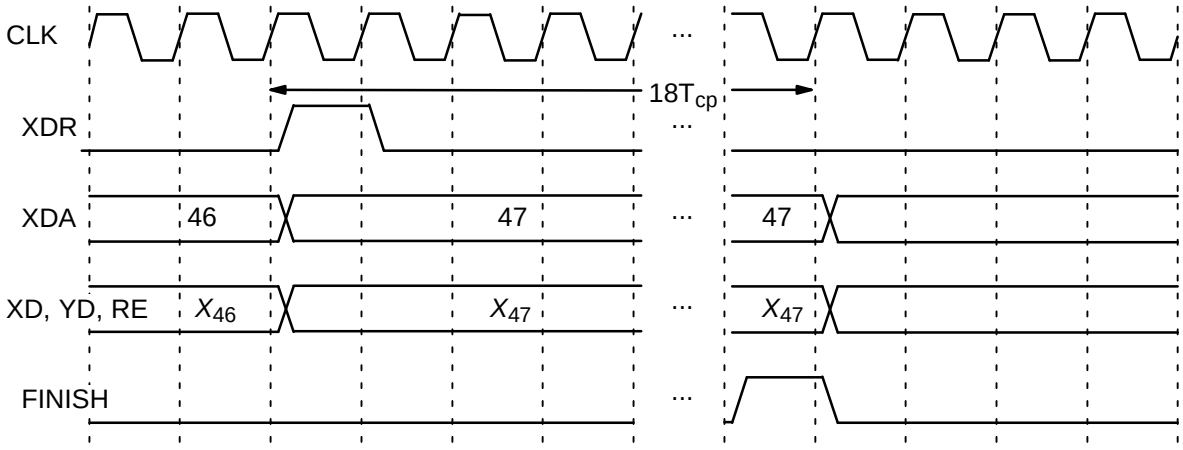

Figure 7: Viterbi Decoder Output Timing (*K* = 48).

$4j+k$, where j varies from 0 to $L/4-1$ and k varies from 0 to 3. The symbol length $L = K+m$ for terminated codes and $L = K$ for tail biting codes.

Table 5: Sign Magnitude Quantisation

Int–eger	Dec–imal	Binary	Range (Min)	Range (Max)
127	127	01111111	126.5	∞
126	126	01111110	125.5	126.5
⋮	⋮	⋮	⋮	⋮
2	2	00000010	1.5	2.5
1	1	00000001	0.5	1.5
0	0	00000000	–0.5	0.5
0	128	10000000	–0.5	0.5
–1	129	10000001	–1.5	–0.5
–2	130	10000010	–2.5	–1.5
⋮	⋮	⋮	⋮	⋮
–126	254	11111110	–126.5	–125.5
–127	255	11111111	–∞	–126.5

Table 6: Two's Complement Quantisation

Int–eger	Dec–imal	Binary	Range (Min)	Range (Max)
127	127	01111111	126.5	∞
126	126	01111110	125.5	126.5
⋮	⋮	⋮	⋮	⋮
2	2	00000010	1.5	2.5
1	1	00000001	0.5	1.5
0	0	00000000	–0.5	0.5
–1	255	11111111	–1.5	–0.5
–2	254	11111110	–2.5	–1.5
⋮	⋮	⋮	⋮	⋮
–126	130	10000010	–126.5	–125.5
–127	129	10000001	–127.5	–126.5
–128	128	10000000	–∞	–127.5

Table 7: Rate 2/3 input data format

Data	1st input	2nd input
R0I	Y0($4j$)	Y0($4j$+2)
R1I	Y1($4j$)	Y1($4j$+2)
R2I	Y1($4j$+1)	Y0($4j$+3)

The received data should be stored in an (L/2)x24 memory. The read address RA is internally divided by 2 so that the correct address is output. To ensure correct operation R0I, R1I and R2I must be held during the 36 clock cycles before changing, using RR and RA to read the data from a synchronous read input memory.

For example, if K = 192 and TB = 0, then RA changes from 0 to (192+10)/2–1 = 100. The input data is stored in a 101x24 memory.

Mode Selection

To minimise the decoder complexity, the MODE input can be used to select only those rates that are expected to be used. When MODE is low, the decoder can decode rate 1/2 and 1/3 codes. When MODE is high, the decoder can decode rate 1/2, 1/3, and 1/4 codes. MODE should only be connected to GND or VCC. Connecting MODE to an input pin or logic will result in excessive logic and reduced decoding speed.

Serial Operation

When the SERIAL input is high, the decoder uses an internal serial to parallel converter to convert serially received data, for example in BPSK modulation, into parallel received data. The received clock should be input to START. This clock must not be divided down and must be equal to the received symbol rate. The received data must be valid from one to two CLK cycles after the rising edge of START and be input to R0I only.

The data corresponding to Y0 to Y3 is assumed to be received in this order. For example for rate 1/2, the data for Y0 is received first, followed by the data for Y1.

Note that FINISH only goes high at the end of each received code symbol. This consists of n received data symbols for a rate 1/n code. Due to the serial to parallel operation, the decoder delay increases by n–1 received data periods.

Automatic Synchronisation

When BLK = 0, KN = 0 and SYNC_EN = 1, automatic synchronisation to the coded symbol is enabled. This counts the number of state metric normalisations within the decoder. If the count exceeds the synchronisation threshold (SYNC_TH) before the end of the synchronisation period (SYNC_PD), SYNC_OUT will go high for one code symbol period. The normalisation counter is then disabled for one SYNC_PD, to allow the decoder to settle to its new synchronisation state. A new count is then started. If the threshold is not exceeded at the end of the SYNC_PD, the normalisation and period counters are reset, and a new count is started.

When SYNC_EN is high, SYNC_OUT is internally used by the decoder to change the synchronisation state. For serial operation, the code symbol period is increased by one received data period. This is performed only once, and causes the serial to parallel conversion to load one received

data period later. With rate 1/2 Gray mapped QPSK operation, the received data is rotated by 0 or 90°, depending on the synchronisation state (0 or 1).

Note that if SYNC_EN is low, SYNC_OUT is not disabled (however, the internal synchronisation state is not allowed to change). This allows SYNC_OUT to be externally used to control the synchronisation state of an external synchronisation circuit.

With rate 1/2 operation at an E_b/N_0 of 4.2 dB (corresponding to a BER of 8.3×10^{-6} with DELAY = 1), the state metric normalisation rate is about 6.7×10^{-3}. When the decoder is out is sync though, the normalisation rate increases to about 4.7×10^{-2}. Thus, with a SYNC_PD of 128, a SYNC_TH of $128 \times 4.7 \times 10^{-2} = 6$ should provide a robust threshold. The average count when in sync is less than one. This should ensure that the decoder does not lose sync when it is in sync and that it quickly synchronises when out of sync.

Other inputs

The decode chip select (DCS) input is used to put the Viterbi decoder in a low power mode when low. Note that this is not a clock enable input. The decoder state is lost when DCS goes low.

The RST input when high synchronously forces all flip–flops low. This is useful for VHDL simulations where flip–flops are initially in an unknown state. The decoded output will be unknown until the unknown data in RAM is flushed out. The length of the unknown output data should be equal to the decoder delay.

Simulation Software

Free software for simulating the VA10V Viterbi decoder in additive white Gaussian noise (AWGN) is available by sending an email to info@sworld.com.au with “va10vsim request” in the subject header. The software uses an exact functional simulation of the VA10V Viterbi decoder, including all quantisation and limiting effects.

Figure 8 shows the AWGN performance with binary modulation obtained for the rate 1/2 (A = 45), 1/3 (A = 40) and 1/4 (A = 35) convolutional codes decoded by the VA10V Viterbi decoder with DELAY = 1 and BLK = 0.

Figures 9, 10 and 11 gives frame error rate (FER) performance for the VA10V decoder for data lengths of 48, 96 and 192 bits, respectively, as specificied in the 3GPP2 standard. An AWGN channel and binary modulation is used.

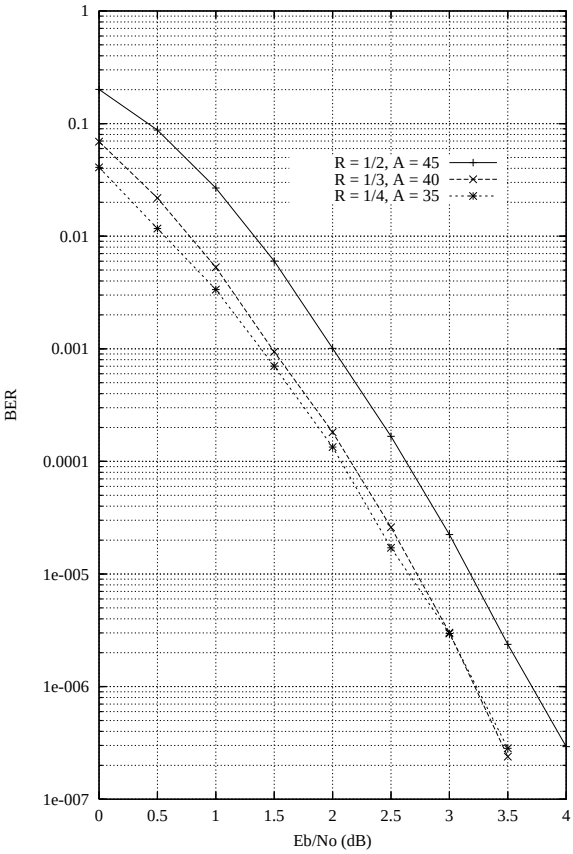

Figure 8: Rate 1/2, 1/3 and 1/4 convolutional code decoded by VA10V performance.

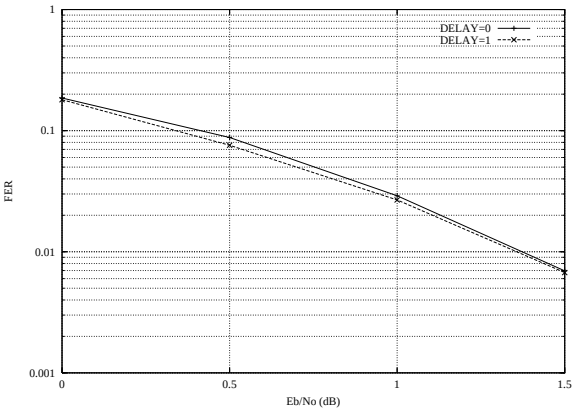

Figure 9: FER performance, K = 48, rate 1/4 and tailbiting.

Ordering Information

SW–VA10V–SOS (SignOnce Site License)
SW–VA10V–SOP (SignOnce Project License)
SW–VA10V–VHD (VHDL ASIC License)

All licenses include EDIF and VHDL cores. The VHDL cores can only be used for simulation in the SignOnce licenses. The SignOnce and ASIC licenses allows unlimited instantiations.

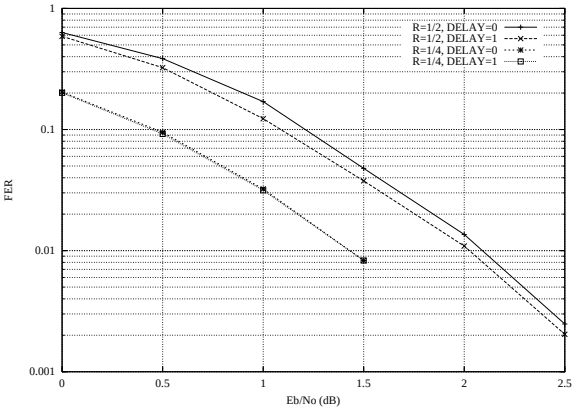

Figure 10: FER performance,, *K* = 96, rate 1/2 and 1/4 and tailbiting.

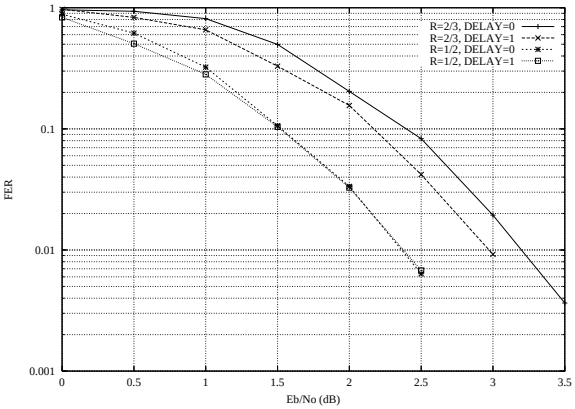

Figure 11: FER performance, *K* = 192, rate 2/3 and 1/2 and terminated.

Note that *Small World Communications* only provides software and does not provide the actual devices themselves. Please contact *Small World Communications* for a quote.

References

[1] A. J. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Trans. Inform. Theory*, vol. IT–13, pp. 260–269, Apr. 1967.

[2] Third Generation Partnership Project 2 (3GPP2), “Physical layer for cdma2000 extended cell high rate packet data air interface specification,” 3GPP2 C.S0098–200–0 Version 1.0, Sep. 2011.

[3] P. J. Lee, “Further results on rate 1/*N* convolutional code constructions with minimum required SNR criterion,” *IEEE Trans. on Commun*., vol. COM–34, pp. 395–399, Apr. 1986.

Small World Communications, 6 First Avenue, Payneham South SA 5070, Australia.
info@sworld.com.au ph. +61 8 8332 0319
http://www.sworld.com.au fax +61 8 8332 3177

Version History

- 0.00 18 September 2013. Preliminary product specification.
- 1.00 23 September 2013. First release. Added BER simulation results and updated FER simulation results.
- 1.01 28 October 2013. Corrected minimum tail–biting data lengths.
- 1.02 30 October 2013. Added STDEL input.