VA06C 64 State 4x8PSK CCSDS/ECSS Viterbi Decoder

4 July 2023 (Version 1.02) Product Specification

VA06C Features

- 64 state 4x8PSK CCSDS and ECSS compatible Viterbi decoder
- Bandwidth efficiencies of 2, 2.25, 2.5 or 2.75 bit/sym
- Up to 293 MHz 4x8PSK symbol rate
- Up to 3.2 Gbit/s data rate
- Global clock enable
- 6–bit received I and Q data
- Optional differential decoder for phase ambiguity resolution
- Optional automatic 4x8PSK symbol synchronisation
- Estimated symbol error output
- Minimum traceback depth of 33 or 65
- 12,500 6–input LUTs
- Asynchronous logic free design
- Available as EDIF and VHDL core for Xilinx FPGAs under SignOnce IP License. ASIC, Intel/Altera, Lattice and Microsemi/Actel cores available on request.

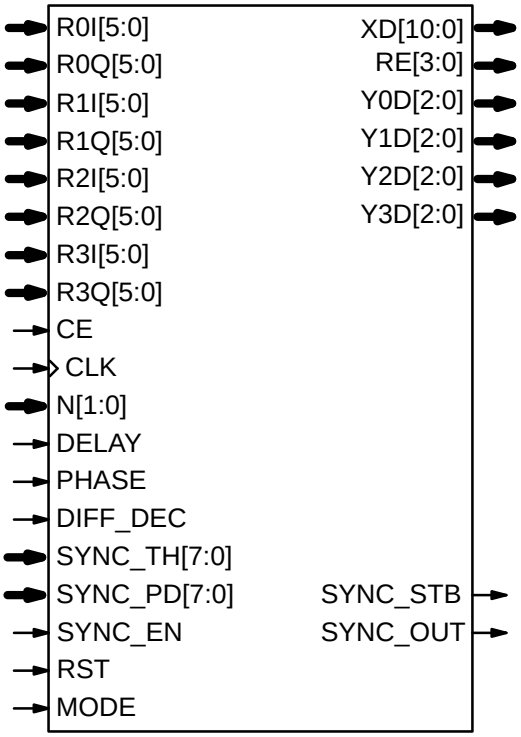

Figure 1: VA06C schematic symbol.

Introduction

The VA06C is a fully parallel 64 state error control decoder using the maximum likelihood Viterbi algorithm for the 4x8PSK trellis codes specified in the CCSDS [1] and ECSS [2] standards. The trellis code uses a multi–dimensional 4x8PSK signal set with a systematic recursive convolutional encoder with three checked bits and six delay elements. By choosing which bits are checked by the convolutional encoder and reducing the number of unchecked bits, bandwidth efficiencies of either 2, 2.25, 2.5 or 2.75 bit/sym can be selected. The CCSDS standard specifies all four of these efficiencies, while ECSS only specifies 2 and 2.5 bit/sym.

Figure 1 shows the schematic symbol for the VA06C decoder. The EDIF core can be used with Xilinx Foundation or Integrated Software Environment (ISE) software. The VHDL core can be used with Xilinx ISE or Vivado software. Custom VHDL cores can be used in ASIC designs.

Table 1 shows the performance achieved with various Xilinx parts. T_{cp} is the minimum clock period over recommended operating conditions with CE = 1 and MODE = 1. These performance figures may change due to device utilisation and configuration. Note that Zynq devices up to XC7Z020 and from XC7Z030 use programmable logic equivalent to Artix–7 and Kintex–7 devices, respectively

Signal Descriptions

Note that the MODE[1:0] inputs are used for selecting an implementation configuration of the decoder. These inputs should only be tied to VCC or GND. They should not be connected to logic or input pins.

CE	Clock Enable
CLK	System Clock
DIFF_DEC	Differential Decoder Enable
	0 = disabled
	1 = enabled
DELAY	Decoder Delay Select
	0 = 138
	1 = 266
MODE	Delay Implementation
	0 = Short Delay Only
	1 = Short and Long Delay

Table 1: Performance of Xilinx parts.

Xilinx Part	T_{cp} (ns)	f_d (Mbit/s)	
		2 bit/sym	2.75 bit/sym
XC7S75T–1	11.748	680	936
XC7S75T–2	9.606	832	1145
XC7A75T–1	11.832	676	929
XC7A75T–2	9.662	827	1138
XC7A75T–3	8.595	930	1279
XC7K70T–1	8.012	998	1372
XC7K70T–2	6.467	1237	1700
XC7K70T–3	5.974	1339	1841
XCKU035–1	5.925	1350	1856
XCKU035–2	4.972	1609	2212
XCKU035–3	4.399	1818	2500
XCKU3P–1	4.301	1860	2557
XCKU3P–2	3.680	2173	2989
XCKU3P–3	3.407	2348	3228

N	Bandwidth Efficiency 0 = 2 bit/sym 1 = 2.25 bit/sym 2 = 2.5 bit/sym 3 = 2.75 bit/sym
PHASE	8PSK Phase Offset Select 0 = 0° 1 = 22.5°
R0I–R3I	Inphase Received Data
R0Q–R3Q	Quadrature Received Data
RE	Estimated Symbol Error
RST	Synchronous Reset
SYNC_EN	Synchronisation Enable
SYNC_OUT	Synchronisation Output SIgnal
SYNC_PD	Synchronisation Period (1 to 255)
SYNC_STB	Synchronisation Strobe
SYNC_TH	Synchronisation Threshold (1 to 255)
XD	Decoded Data Output
Y0D–Y3D	Decoded Symbol Output

Multi–Dimensional Trellis Code

Figure 2 gives a block diagram of the 64 state rate 3/4 systematic encoder used by the 4x8PSK multi–dimensional trellis code.

The encoder polynomials are defined as

$$g_0(D) = D^6 + D + 1 \equiv 103_8 \quad (1)$$
$$g_1(D) = D^2 + D \equiv 006_8 \quad (2)$$
$$g_2(D) = D^4 + D^2 \equiv 024_8 \quad (3)$$
$$g_3(D) = D^5 + D^3 \equiv 050_8 \quad (4)$$

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., $g_0 = 103_8 = 1000011_2 \equiv g_0(D) = D^6 + D + 1$.

The signal set mapper is described by

$$\begin{bmatrix} z^0 \\ z^1 \\ z^2 \\ z^3 \end{bmatrix} = \sum_{p=0}^{11} y^p t^p \pmod 8 \quad (5)$$

where t^p are the cosets given in Table 2 and y^p are the binary inputs to the signal set mapper. Figure 3 shows the signal set mapper with the corresponding inputs to obtain the various code rates (selected by input N[1:0]). Three bit modulo–8 adders are used.

Table 2: 4x8PSK Cosets

p	0	1	2	3	4	5	6	7	8	9	10	11
t^p	0 0 0 1	0 0 1 1	0 1 0 1	0 0 0 2	1 1 1 1	0 0 2 2	0 2 0 2	0 0 0 4	2 2 2 2	0 0 4 4	0 4 0 4	4 4 4 4

To resolve 45° phase ambiguities in the 8PSK signal set, a differential encoder is used. Figure 4 shows the differential encoder where w^p are the differential encoder inputs. As only bits y^4, y^8 and y^{11}, are affected by a phase rotation (corresponding to cosets with all 1's, 2's and 4's, respectively), these are the only bits checked by the differential encoder. The corresponding differential decoder within the core can be disabled by setting DIFF_DEC = 0.

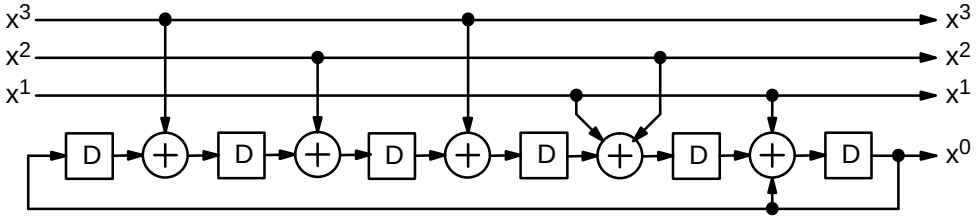

Figure 2: 64 state rate 3/4 systematic convolutional encoder.

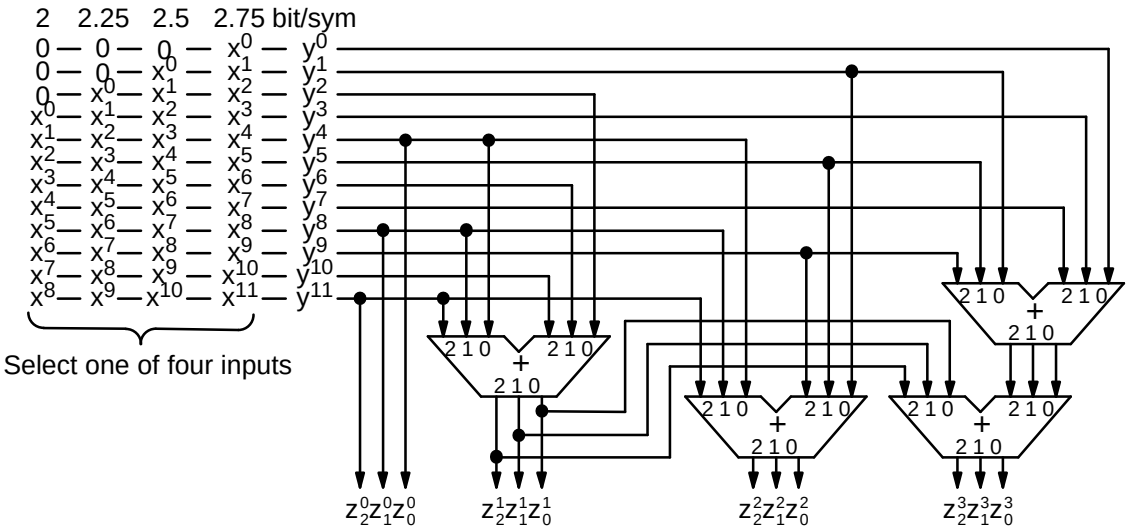

Figure 3: 4x8PSK Signal Set Mapper

Viterbi Decoder

The Viterbi decoder is designed to be flexible. However, It can be operated in continuous mode only.

Theory of Operation

The Viterbi decoding algorithm [3] finds the most likely transmitted sequence given the received noisy sequence.

For eight phase shift keying (8PSK) modulation the received signal in the complex domain at time *k* is described by

$$R_k = A(\exp(j\pi z_k/4 + \phi) + n_k) \qquad (6)$$

where A is the signal amplitude, $z_k \in \{0, 1, \ldots, 7\}$, ϕ is the phase offset (0 or $\pi/8$), and n_k is a Gaussian distributed random variable with zero mean and normalised variance σ^2. Figure 5 shows the signal sets for 8PSK with ϕ = 0 or $\pi/8$ (selected by PHASE = 0 and 1, respectively). We have

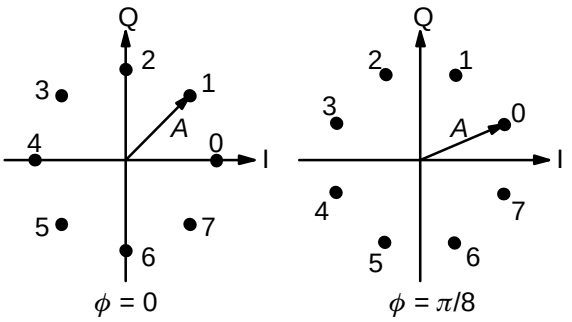

Figure 5: 8PSK with ϕ = 0 or $\pi/8$.

$$\sigma^2 = \left(2K\frac{E_b}{N_0}\right)^{-1} \qquad (7)$$

where E_b/N_0 is the energy per bit to single sided noise density ratio and K is the bandwidth efficiency, either 2, 2.25, 2.5 or 2.75 bit/sym).

Due to quantisation and limiting effects we recommend a value of A = 16 should be used. If A is too small then the quantisation between the signal points becomes too course. If A is too large, then

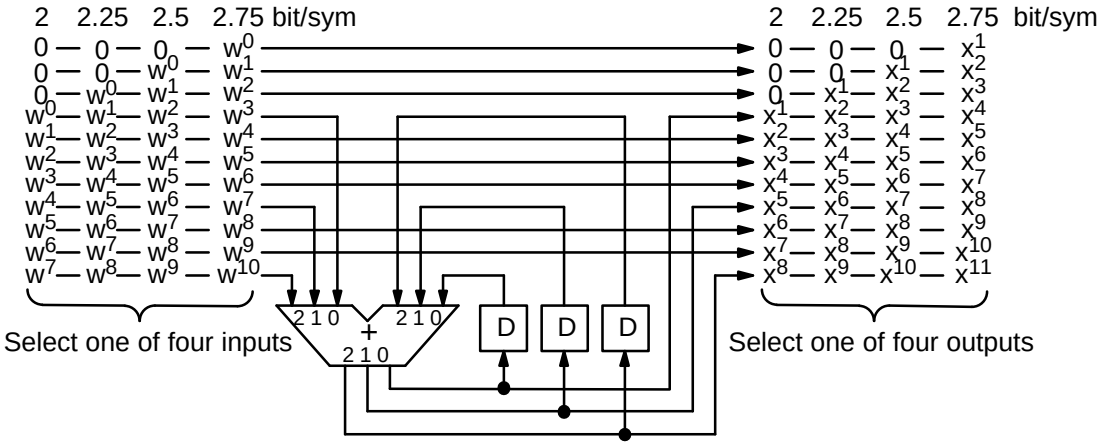

Figure 4: Differential Encoder

the noise values become limited in value, which results in a loss of information to the decoder.

Each received 8PSK received symbol is input as a two dimensional signal with sign magnitude inputs. The 6–bit inputs have 63 quantisation regions with a central dead zone. The quantisation regions are labelled from –31 to +31.

Decoder Operation

The VA06C uses 64 add–compare–select (ACS) circuits in parallel to decode 64 state convolutional codes with three checked bits into the encoder. Each ACS circuit selects the path metric which has the smallest weight out of eight possible paths. The 3–bit path decisions for each state is stored in a memory of depth L, where L is 32 or 64, depending on the value of DELAY. The path decisions in this memory are output in reverse time order. Using the state with the minimum state metric (SM), the traceback is performed over L clock cycles.

As the initial L tracebacks are not reliable, the time reversed path decisions are delayed a further $2L$ clock cycles using a depth $2L$ memory and a second traceback is started using the final state of the first traceback. The L 3–bit path decisions (x^1 to x^3) and the least significant bit of the state (x^0) are then time reversed using a depth L memory. Thus, the total delay of the traceback is $4L$. An additional 10 clock cycles are used in various pipelines to give a total delay of $4L$+10.

If DELAY is tied low, then MODE[1] should also be tied low to reduce the amount of memory used in the decoder. If DELAY is high or tied to logic, then MODE[1] should be tied high.

To calculate the branch metrics (BM) we first need to calculate the 2–D BM for each of the eight points in the 8PSK signal set. For a 2–D received point (x,y) we have the squared Euclidean distance to a signal point (a,b) as

$$\begin{aligned} d^2 &= (x-a)^2 + (y-b)^2 \\ &= x^2 - 2ax + a^2 + y^2 - 2by + b^2. \end{aligned} \tag{8}$$

In this case x represents the in–phase (I) component of the received signal and y represents the quadrature (Q) component. For MPSK modulation we have

$$\begin{aligned} a &= A\cos(\theta) \\ b &= A\sin(\theta) \end{aligned} \tag{9}$$

where A is the constant signal amplitude and θ represents the modulated phase. This gives

$$\begin{aligned} d^2 &= x^2 + y^2 + A^2 \\ &\quad - 2A(cos(\theta)x + \sin(\theta)y). \end{aligned} \tag{10}$$

The terms x^2, y^2 and A^2 are all constant for each received value. As they do not affect the decision they can thus be subtracted. We can also divide by $2A$ without affecting the decision, giving the BM

$$\text{BM} = -\cos(\theta)x - \sin(\theta)y. \tag{11}$$

That is, the 2–D BM calculator (BMC2) does not need to know the amplitude of the transmitted signal.

Since none of the most significant bits of the 8PSK signal set are checked by the convolutional encoder, BMC2 finds the smallest BM for the path decision points {0,4}, {1,5}, {2,6} and {3,7}. The 2–D BMs are adjusted and normalised so that the minimum 2–D BM is always 0 and the maximum possible value is 62.

For 2, 2.25, 2.5 or 2.75 bit/sym, there are 16 different sets of 32, 64, 128 or 256 parallel paths, respectively, along each path in the code trellis. For each set of parallel paths, the path closest to the received signal must be chosen, giving 16 sets of path decisions of 5, 6, 7 or 8 bits, respectively. The 16 BMs for the selected path are passed to the state metric calculator. The maximum multi–D BM is 124 (requiring 7–bit quantisation) and 7–bit state metrics are used.

To reduce the complexity of calculating the multi–D BMs, a length four time varying trellis is used [4]. The number of states from time 0 to 3 are {4,8,16,16} for 2, 2.5 and 2.75 bit/sym and {4,16,16,16} for 2.25 bit/sym. Note that there is only one initial state. The number of paths into each state are {1,2,2,4}, {1,2,2,2}, {1,1,2,2} and {1,1,1,2} for 2.75, 2.5, 2.25 and 2 bit/sym, respectively. By selecting the appropriate paths, a single circuit can be used to implement all four code rates.

The path decision bits from this trellis along with the path decisions from BMC2 are then delayed and used with the four decoded bits from the traceback circuit. These bits are used to determine the decoded bits of the parallel paths, along with the decoded symbol outputs.

XD[10:0] corresponds to decoded bits of w^{10} down to w^0. Note that bits XD[10:8], XD[10:9] and XD[10] are equal to zero for 2, 2.25 and 2.5 bit/sym, respectively.

Y0D[2:0] to Y3D[2:0] corresponds to the decoded symbols of z^0 to z^3, respectively.

RE[3:0] is obtained by comparing the symbol hard decision of the input data (appropriately delayed) with the re–encoded output symbols. For example, say the hard decision for R0I[5:0] and R0Q[5:0] is 5 and Y0D[2:0] is 4. Then RE[0] = 1, indicating an error in the first symbol. At low BER,

these outputs can be used to give a good estimate of the channel symbol error rate.

Data Format

The decoder uses six bit signed magnitude quantisation for R0I to R3I and R0Q to R3Q. Table 3 shows the six bit quantisation ranges. Note that 0 and 32 indicate the central dead zone and have the same range. Also, most analog to digital to digital (A/D) circuits do have a central dead zone. For maximum performance, we recommend that seven bit A/Ds are used with the output converted to six bits so that the appropriate ranges are obtained.

For input data quantised to less than 6–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 4–bit received data R0IT[3:0], where R0IT[3] is the sign bit, we have R0I[5:2] = R0IT[3:0] and R0I[1:0] = 2 in decimal (10 in binary).

Table 3: Quantisation for R0I to R3I and R0Q to R3Q.

Decimal	Binary	Range
31	011111	30.5↔ ∞
30	011110	29.5↔30.5
⋮	⋮	⋮
2	000010	1.5↔2.5
1	000001	0.5↔1.5
0	000000	–0.5↔0.5
32	100000	–0.5↔0.5
33	100001	–1.5↔–0.5
34	100010	–2.5↔–1.5
⋮	⋮	⋮
62	111110	–30.5↔–29.5
63	111111	– ∞ ↔–30.5

Other inputs

The RST input when high synchronously resets all flip–flops (FF) on the low to high transition of CLK. This is useful for VHDL simulations where flip–flops are initially in an unknown state.

The CE is a global clock enable and will hold all flip–flops and memories when low. The setup time for CE is at least one half that given in Table 1. For example, when CE is high for at least every other clock cycle, the internal clock frequency can be doubled.

Automatic Synchronisation

The automatic synchronisation circuit accumulates the state metric normalisations within the decoder. As normalisation is pipelined for one clock cycle, the maximum normalisation rate is 0.5.

If the normalisation count exceeds the synchronisation threshold (SYNC_TH) before the end of the synchronisation period (SYNC_PD), SYNC_OUT and SYNC_STB will go high for one clock cycle. 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 SYNC_PD, SYNC_STB will go high for one clock cycle (SYNC_OUT stays low), the normalisation and period counters are reset, and a new count is started.

When SYNC_EN is high, the decoder will automatically synchronise to the received 4x8PSK signal set. This is performed by delaying selected input symbols by one clock cycle and selecting the appropriate inputs, depending on one of four internal synchronisation states (SS). The SS is incremented every time that SYNC_OUT goes high. Note that SYNC_STB and SYNC_OUT always operate regardless of SYNC_EN. When SYNC_EN is low, the inputs are directly used without any shift, that is the SS is ignored.

Table 4 gives the normalisation rates for the four code rates when the decoder is synchronised and not synchronised for PHASE = 0 and A = 16. Taking the mean of these two values and multiplying by a SYNC_PD of 128 gives the SYNC_TH values shown. Due to the small SYNC_TH value for 2.75 bit/sym, a longer SYNC_PD might need to be used. For PHASE = 1, the normalisation rates should be multiplied by 1.3.

Table 4: Synchronisation values for PHASE = 0 and A = 16

K (bit/sym)	E_b/N_0 (dB)	Norm Rate (in sync)	Norm Rate (out of sync)	Threshold (period = 128)
2	6.0	0.0622	0.249	20
2.25	6.5	0.0357	0.159	13
2.5	7.5	0.0135	0.108	8
2.75	8.5	0.0040	0.0393	3

Simulation Software

Free software for simulating the VA06C Viterbi decoder in additive white Gaussian noise (AWGN) is available by sending an email to

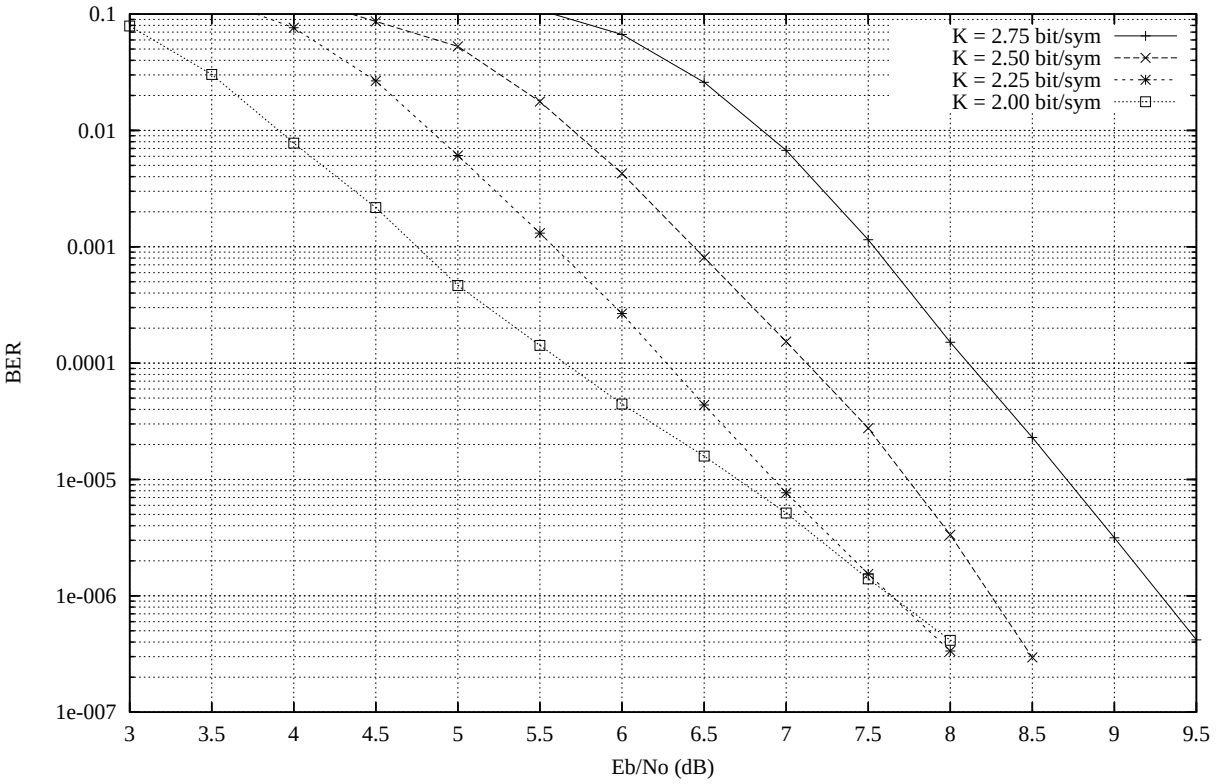

Figure 6: BER performance for AWGN channel, DELAY=1, DIFF_DEC=1, PHASE=1 and A = 16.

info@sworld.com.au with "va06csim request" in the subject header. The software uses an exact functional simulation of the VA06C Viterbi decoder, including all quantisation and limiting effects. The software can also be used to encode and decode external data.

Figure 6 gives the bit error rate (BER) versus energy per bit to single–sided noise density ratio (E_b/N_0) performance of the VA06C Viterbi decoder for all four code rates, additive white Gaussian noise (AWGN), DELAY = 1, DIFF_DEC = 1, PHASE = 1 and A = 16. Each simulation point counted 500 symbol errrors.

Curiously, the performance for 2 bit/sym is worse than 2.25 bit/sym at low BERs. We can understand this by examining the normalised free squared Euclidean distance (d^2_{free}), number of nearest neighbours (N_{free}) and the asymptototic coding gain compared to uncoded QPSK, $\gamma = 10\log_{10}(Kd^2_{\text{free}}/4)$, as given Table 5.

As d^2_{free} is the same for both 2 and 2.25 bit/sym, then at infinite E_b/N_0 we can expect that 2.25 bit/sym to have 0.51 greater coding gain than 2 bit/sym. Similarly, we can expect 2.5 bit/sym to have 0.97 dB greater asymptotic coding gain than 2 bit/sym. At low E_b/N_0, the smaller values for N_{free} imply that 2 bit/sym will perform better.

Table 5: Asymptotic coding gains

K (bit/sym)	d^2_{free}	N_{free}	γ (dB)
2	4.0	4	3.01
2.25	4.0	12	3.52
2.5	4.0	28	3.98
2.75	2.343	8	2.07

Ordering Information

SW–VA06C–SOS (SignOnce Site License)
SW–VA06C–SOP (SignOnce Project License)
SW–VA06C–VHD (VHDL ASIC License)

All licenses include EDIF and VHDL cores. The SignOnce and ASIC licenses allows unlimited instantiations. The EDIF core can be used for Virtex–2, Spartan–3 and Virtex–4 with Foundation or ISE software. The VHDL core can be used for Virtex–5, Spartan–6, Virtex–6, 7–Series, UltraScale and UltraScale+ with ISE or Vivado software.

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] Consultative Committee for Space Data Systems, “Radio frequency and modulation systems Part 1: Earth stations and spacecraft,” CCSDS 401.0–B–21, July 2011.

[2] European Cooperation for Space Standardization, “Space engineering: Radio frequency and modulation,” ECSS–E–ST–50–05C Rev. 1, 5 Mar. 2009.

[3] 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.

[4] S. S. Pietrobon, R. H. Deng, A. Lafanechére, G. Ungerboeck and D. J. Costello, Jr., “Trellis–coded multidimensional phase modulation,” *IEEE Trans. Inform. Theory*, vol. 36, pp. 63–89, Jan. 1990.

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 7117 1416

Version History

- 0.00 9 Jan. 2012. VA06C preliminary product specification.
- 0.01 11 Jan. 2012. Added AR input. Updated CCSDS reference.
- 0.02 13 Jan. 2012. Added SYNC_STB output.
- 0.03 26 Mar. 2012. Deleted A[5:0] input. Added MODE[1:0] and SYNC_EN inputs. Added Virtex–4 performance. Added more detail to Decoder Operation.
- 1.00 12 Apr. 2012. Updated Virtex–4 performance and complexity. Added Spartan–3, Spartan–6, Virtex–5, Virtex–6 and Kintex–7 performance and complexity. Added synchronisation values and recommended amplitude. Added BER performance.
- 1.01 15 Jun. 2023. Deleted AR input. Changed MODE[1:0] to MODE. Deleted Spartan–3, Spartan–6, Virtex–4, Virtex–5 and Virtex–6 performance and complexity. Added Spartan–7, Artix–7, UltraScale and UltraScale+ performance. Updated complexity.
- 1.02 4 July 2023. Replaced Performance section with Simulation Software section. Added availability of free BER simulation software.