



PCD03VH Features

Turbo Decoder

- 8 state 3GPP™ LTE and 3GPP2 1xEV-DO Release B compatible
- Rate 1/2, 1/3, 1/4 or 1/5
- 40 to 6144 (3GPP™ LTE) or 17 to 20730 (3GPP2) bit interleaver
- Includes ping-pong input and output memories.
- Optional external interleaver parameters for 3GPP™ LTE and programmable row coefficients for 3GPP2 interleaver
- Up to 107 MHz internal clock
- Up to 10.2 Mbit/s with 5 decoder iterations
- 6-bit signed magnitude or two's complement input data with serial input
- Optional log-MAP or max-log-MAP constituent decoder algorithms
- Up to 128 iterations in 1/2 iteration steps.
- Optional power efficient early stopping
- Optional extrinsic information scaling and limiting
- 16 or 24 bit CRC check
- Implement one or two different standards from the one core
- Free simulation software
- Available as EDIF core and VHDL simulation core for Xilinx Virtex-II Pro, Spartan-3, Virtex-4 and Virtex-5 FPGAs under SignOnce IP License. Actel and Lattice FPGA cores available on request.
- Available as VHDL core for ASICs
- Low cost university license also available

Introduction

The PCD03VH is a fully compatible 3GPP™ LTE [1] and 3GPP2 1xEV-DO Release B [2] error control decoder. The 3GPP™ and 3GPP2 turbo codes have a number of similarities, but also important differences which affect their implementation. The biggest similarity is that they use the same constituent 8 state systematic recursive convolutional code. The 3GPP2 interleaver uses a row/column architecture. 3GPP™ LTE uses a simple quadratic permutation interleaver. The 3GPP™ and 3GPP2 codes also have different tails.

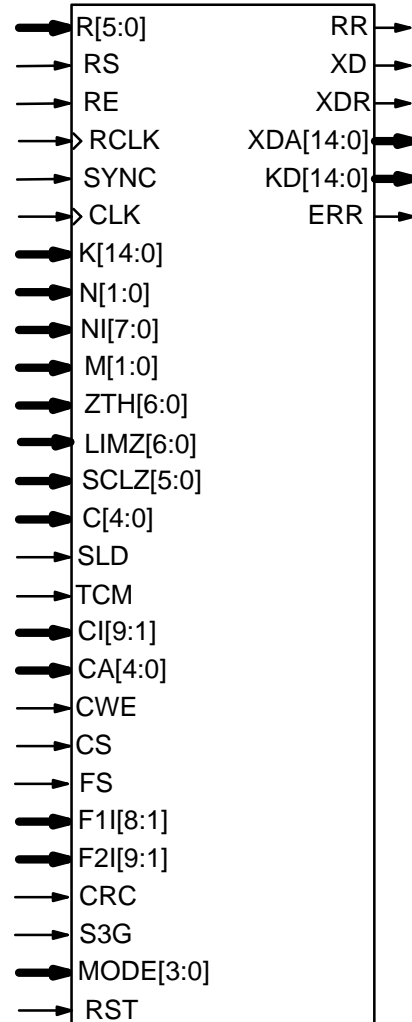


Figure 1: PCD03VH schematic symbol.

For 3GPP2, the use of powers of 2 greatly simplifies parameter calculation. The block length can range from 17 to 21504 bits. The forward link for 1xEV-DO uses eight different interleavers (122, 250, 506, 1018, 2042, 3066, 4090, and 5114 bits). The reverse link for 1xEV-DO uses 12 different interleavers (122, 250, 506, 762, 1018, 1530, 2042, 3066, 4090, 6138, 8186, and 12282 bits). For interleaver sizes less than 65, the parameter n is fixed at 2 (4 columns).

For 3GPP™ LTE, there are 188 interleaver sizes ranging from 40 to 6144 bits. Two parameters f_1 and f_2 are used by the interleaver. All interleaver sizes from 40 to 504 bits that are a multiple

Table 1: Resources used.

Configuration	Turbo Rates	Virtex-4 Slices	Virtex-5 Slices	18KB Block RAMs
3GPP™ LTE/3GPP2 1xEV-DO Turbo (max-log-MAP)	1/2-1/5	2743	2030	51
3GPP™ LTE/3GPP2 1xEV-DO Turbo (small log-MAP)	1/2-1/5	3145	2313	51
3GPP™ LTE/3GPP2 1xEV-DO Turbo (large log-MAP)	1/2-1/5	3387	2348	51
3GPP™ LTE Turbo (small log-MAP)	1/3	2305	1801	17
3GPP2 1xEV-DO Turbo (small log-MAP)	1/2-1/5	2719	2079	51

of 8, 512 to 1008 bits that are a multiple of 16, 1024 to 2016 bits that are multiple of 32, and 2048 to 6144 bits that are a multiple of 64 are specified.

For 3GPP™ only a code rate of 1/3 is specified. However, the PCD03VH can also optionally decode rate 1/2, 1/4 and 1/5 turbo codes. For 3GPP2, rate 1/2, 1/3, 1/4 and 1/5 are specified.

For 3GPP™, each tail of the two constituent encoders are terminated using all the data and parity bits (for a total of 12 bits for rate 1/3). For rate 1/2 the PCD03VH uses the same tail as for rate 1/3. For rate 1/4 and 1/5, a total of 18 tail bits are used.

For 3GPP2, the number of tail bits is equal to $6n$, for a rate $1/n$ code. For rate 1/2, the tails for 3GPP™ and 3GPP2 are determined in the same way. For rate 1/3 and 1/4 though, data bits are repeated to make up for the additional tail bits in the 3GPP2 standard. For rate 1/5 the data and second parity bit are each repeated two times for 1xEV-DO.

The MAP03V MAP decoder core is used with the PCD03VH core to iteratively decode the 3GPP™ or 3GPP2 turbo code. The Log-MAP algorithm for maximum performance or the max-log-MAP algorithm for minimum complexity can be selected. The sliding block algorithm is used with sliding block lengths of 32 or 64. Six-bit quantisation is used for maximum performance. The extrinsic information can be scaled and limited with each half iteration, improving performance with max-log-MAP decoding.

The turbo decoder can achieve up to 10.2 Mbit/s with 5 iterations using a 107 MHz internal clock ($K = 5114$). Optional early stopping allows the decoder to greatly reduce power consumption with little degradation in performance.

Figure 1 shows the schematic symbol for the PCD03VH decoder. The EDIF core can be used with Xilinx Integrated Software Environment (ISE) software to implement the core in Xilinx FPGA's. The VHDL core can be used in ASIC designs.

Table 1 shows the resources used for Virtex-4 devices. The number of slices may slightly vary

with Virtex-IIP, Spartan-3 and Virtex-5 devices. The MODE[3:0] inputs can be used to select various decoder implementations. The input/output memory is not included. Only one global clock is used. No other resources are used. The RAMs refer to BlockRAMs. Turbo* indicates small log-MAP.

Table 2: Performance of Xilinx parts.

Xilinx Part	T_{cp} (ns)	Turbo* Mbit/s
XC3S400-4	23.337	4.07
XC3S400-5	20.393	4.66
XC2VP4-5	17.030	5.58
XC2VP4-6	15.237	6.23
XC2VP4-7	13.100	7.25
XC4VLX25-10	14.042	6.77
XC4VLX25-11	11.975	7.93
XC4VLX25-12	10.698	8.88
XC5VLX30-1	12.038	7.89
XC5VLX30-2	10.271	9.25
XC5VLX30-3	9.298	10.22

*small log-MAP, 5 iterations, $K = 5114$

Table 2 shows the performance achieved with various Xilinx parts. T_{cp} is the minimum clock period over recommended operating conditions. These performance figures may change due to device utilisation and configuration.

Signal Descriptions

C	MAP Decoder Constant 0-9 (MODE1 = 0) 0-17 (MODE1 = 1)
CA	3GGP2 row constant address (0 to 31)
CI	3GPP2 row constant (0 to 511)
CRC	Cyclic Redundancy Check select $0 \rightarrow g(D) = 1 + D^5 + D^{12} + D^{16}$ $1 \rightarrow g(D) = 1 + D + D^5 + D^6 + D^{23} + D^{24}$
CS	3GPP2 row constant select 0 = Select internal row constants 1 = Select programmed row constants

CWE	3GPP2 row constant write enable
CLK	System Clock
ERR	Cyclic Redundancy Check failure
F1I	LTE external parameter 1 (0 to 255) F1I = f_1 div 2.
F2I	LTE external parameter 2 (0 to 511) F2I = f_2 div 2.
FS	LTE external parameter select 0 = Select internal parameters 1 = Select external parameters
K	Interleaver Length (40–6144 for 3GPP™ LTE, 17–21504 for 3GPP2)
KD	Output Interleaver Length
M	Early Stopping Mode 0 = no early stopping 1 = early stop at odd half iteration 2 = early stop at even half iteration 3 = early stop at any half iteration
MODE	Implementation Mode (see Table 3)
LIMZ	Extrinsic Information Limit (1–127)
N	Code Rate 0 = rate 1/4 ($K \leq 12282$) 1 = rate 1/5 ($K \leq 8186$) 2 = rate 1/2 ($K \leq 20730$) 3 = rate 1/3 ($K \leq 12282$)
NI	Number of Half Iterations (0–255) NI = 2I–1 where I is number of iterations
R	Received Data
RCLK	Received Data Clock
RE	Received Data Enable
RR	Received Data Ready
RS	Received Data Start
RST	Synchronous Reset
S3G	Turbo Decoder Standard Select 0 = 3GPP™ LTE 1 = 3GPP2 1xEV–DO
SCLZ	Extrinsic Information Scale (1–32)
SLD	MAP Decoder Delay 0 = delay 136 1 = delay 264
SYNC	Clock Synchronisation 0 = R, RE, RS synchronous to CLK 1 = R, RE, RS synchronous to RCLK
TCM	Input data type select 0 = Signed Magnitude 1 = Two's Complement
XD	Decoded Data
XDA	Decoded Data Address
XDR	Decoded Data Ready
ZTH	Early Stopping Threshold (1–127)

Table 3 describes each of the MODE[3:0] inputs that are used to select various decoder implementations. Note that MODE[3:0] are “soft” inputs

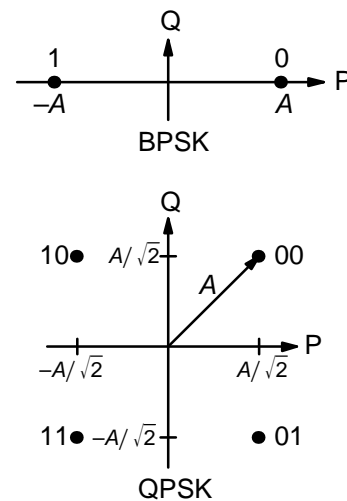


Figure 2: BPSK and QPSK signal sets.

and should not be connected to input pins or logic. These inputs are designed to minimise decoder complexity for the configuration selected.

Table 3: MODE selection

Input	Description
MODE0	0 = max–log–MAP 1 = log–MAP
MODE1	0 = small log–MAP (C4 = 0) 1 = large log–MAP
MODE[3:2]	0 = LTE Implementation 1 = 1xEV–DO Implementation 2 = Not used 3 = LTE and 1xEV–DO Implem.

Turbo Decoder Parameters

For optimal performance, the maximum a posteriori (MAP) [3] constituent decoder is used which is dependent on the signal to noise ratio (SNR). Unlike other turbo decoders with suboptimum soft–in–soft–in (SISO) decoders, using the MAP (or specifically the log–MAP [4]) algorithm can provide up to 0.5 dB coding gain at low SNRs. Log–MAP operation is enabled when MODE0 is high.

With binary phase shift keying (BPSK, $m = 1$) or quadrature phase shift keying (QPSK, $m = 2$) modulation (see Figure 2) the decoder constant C should be adjusted such that

$$C = A\sigma^2 \sqrt{m}/2. \quad (1)$$

where A is the signal amplitude and σ^2 is the normalised noise variance given by

$$\sigma^2 = 1/(2mRE_b/N_0). \quad (2)$$

E_b/N_0 is the energy per bit to single sided noise density ratio and $R = 1/((n+6(\lfloor n/2 \rfloor + 1))/K)$ for

3GPPTM, $R = 1/(n+6n/K)$ for 3GPP2, K is the data length, and $n = 2-5$. C should be rounded to the nearest integer and limited to be no higher than 17 with MODE1 high and 9 with MODE1 low. Max-log-MAP [4] operation occurs when $C = 0$. Due to quantisation effects, $C = 1$ is equivalent to $C = 0$. Max-Log-MAP operation is also enabled when MODE0 is low.

For each code (with a particular block size, rate and number of iterations), there will be a minimum E_b/N_0 where the maximum acceptable BER or FER is achieved. The value of C should be chosen for this E_b/N_0 . This value of C can be kept constant for all E_b/N_0 values for this code. For higher values of E_b/N_0 , there will be negligible degradation in performance, even though C will be higher than optimal [5]. For lower E_b/N_0 values, there could be up to a few tenths of a dB degradation, since C will be lower than optimal. However, this should not have much impact since the BER or FER will already be above the maximum acceptable level anyway.

For fading channels each received value r_k at time k should be scaled by $(A_m\sigma_m^2)/(A_k\sigma_k^2)$ where A_k and σ_k^2 are the no-noise amplitude and normalised variance of r_k and m corresponds to time index of the smallest σ_k^2 . The value of C should be determined by A_m and σ_m^2 . Note that this scaling should be performed for both the log-MAP and max-log-MAP algorithms for optimal performance.

The value of A directly corresponds to the 6-bit signed magnitude inputs (shown in Table 4). The 6-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 $A = 15.7$. This value of A lies in quantisation region 15 (which has a range between 15 and 16).

Since most analogue to digital (A/D) converters do not have a central dead zone, a 7-bit A/D should be used and then converted to 6-bit as shown in the table. This allows maximum performance to be achieved.

For signed magnitude inputs a decimal value of 32 has a magnitude of 0 (equivalent to -0). For two's complement inputs, if 32 is input, this will be internally limited to 33 (equivalent to -31).

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 3-bit received data R0T[2:0], where R0T[2] is the sign bit, we have R0I[5:3] = R0T[2:0] and R0I[2:0] = 4 in decimal (100 in bi-

nary). For punctured input data, all bits must be zero, e.g., R1I[5:0] = 0.

Table 4: Quantisation for R0I, R1I and R2I.

Signed Magnitude		Two's Complement		Range
Dec.	Binary	Dec.	Binary	
31	011111	31	011111	$30.5 \leftrightarrow \infty$
30	011110	30	011110	$29.5 \leftrightarrow 30.5$
⋮	⋮	⋮	⋮	⋮
2	000010	2	000010	$1.5 \leftrightarrow 2.5$
1	000001	1	000001	$0.5 \leftrightarrow 1.5$
0	000000	0	000000	$-0.5 \leftrightarrow 0.5$
33	100001	63	111111	$-1.5 \leftrightarrow -0.5$
34	100010	62	111110	$-2.5 \leftrightarrow -1.5$
⋮	⋮	⋮	⋮	⋮
62	111110	34	100010	$-30.5 \leftrightarrow -29.5$
63	111111	33	100001	$-\infty \leftrightarrow -30.5$

Due to quantisation and limiting effects the value of A should also be adjusted according to the received signal to noise ratio.

Example 1: Rate 1/3 BPSK code operating at $E_b/N_0 = 0.3$ dB. From (2) we have $\sigma^2 = 1.39988$. Assuming $A = 16$ we have from (1) that $C = 11$ to the nearest integer.

The number of turbo decoder half-iterations is given by NI , ranging from 0 to 255. $NI = 2I - 1$ where I is the number of iterations. This is equivalent to 0.5 to 128 iterations. The decoder initially starts at half iteration 0, increasing by one until NI is reached or an earlier time if early stopping is enabled.

The turbo decoder speed f_d is given by

$$f_d = \frac{F_d}{(NI + 1)(1 + (L + 2)/K) + 1/K} \quad (3)$$

where F_d is the CLK frequency and L is the MAP decoder delay in bits (equal to either 136 or 264). The two delays indicate the sliding block length used in the MAP decoder, either 32 or 64, respectively. For short block lengths $L = 136$ should be used to increase decoder speed, while $L = 264$ should be used for larger block sizes to increase performance. This parameter can be selected with the SLD input.

For example, if $F_d = 50$ MHz and $I = 5$ ($NI = 9$) the decoder speed ranges from 1.1 Mbit/s for $K = 40$ and $L = 136$ to 4.7 Mbit/s for $K = 5114$ and $L = 264$.

An important parameter is LIMZ, the limit factors for the extrinsic information. Extrinsic information is the "correction" term that the MAP decoder determines from the received data and a

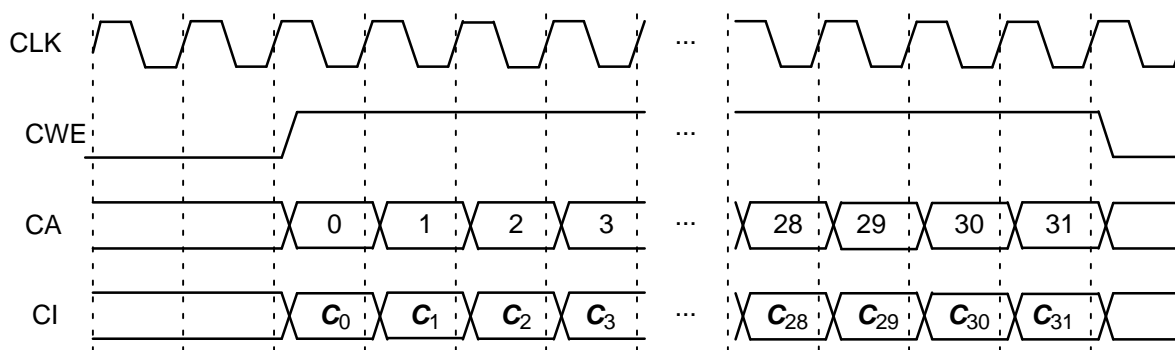


Figure 3: 3GPP2 Parameter Programming.

priori information. It is used as *a priori* information for the next MAP decoding or half iteration. By limiting the correction term, we can prevent the decoder from making decisions too early, which improves decoder performance.

The limit factor LIMZ should vary between 1 and 127, although we recommend that 96 be used.

Another parameter that can be used to adjust decoder performance is SCLZ which ranges from 1 to 32. The extrinsic information is scaled by $SCLZ/32$. Thus, when $SCLZ = 32$, no scaling is performed. For log-MAP decoding we recommend $SCLZ = 29$. For max-log-MAP decoding we recommend $SCLZ = 23$.

There are four decoder operation modes given by M . Mode $M = 0$ decodes a received block with a fixed number of iterations (given by N). Modes 1 to 3 are various early stopping algorithms. Early stopping is used to stop the decoder from iterating further once it has estimated there are zero errors in the block. Mode 1 will stop decoding after an odd number of half-iterations. Mode 2 will stop decoding after an even number of half iterations. Mode 3 will stop after either an odd or even number of half iterations. Further details are given in the next section.

3GPP2 Interleaver Programming

The PCD03VH turbo decoder allows the option of programming the row constants that are used in the 3GPP2 interleaver. There are 32 constants, all of them being odd in value. Note that if K is the interleaver size, the maximum constant value must be less than 2^n , where $n = \lceil \log_2 K \rceil - 5$.

The maximum value of n is 10, so each constant can be represented by a 10 bit value. However, since all the parameters are odd, this implies the least significant bit (lsb) is always equal to one. Thus, only the nine most significant bits (msb) should be input, with the lsb being ignored. This is why the constant input $CI[9:1]$ does not include the lsb CI_0 . For example, if the constant is 349, then

the lsb should be deleted and $\lfloor 349/2 \rfloor = 174$ be input to $CI[9:1]$.

Figure 3 shows an example of programming the 3GPP2 interleaver parameters. During the low to high transition of CLK, if CWE is high, the value at $CI[9:1]$ is programmed into the internal memory at address location $CA[4:0]$.

Note that K or CS can be changed in any order before, during or after programming the row constants. As long as the correct K and CS are input to the decoder before decoding begins, the decoder will use the selected parameters. If CS is low, the internal 3GPP2 standard parameters are selected. If CS is high, the externally programmed parameters are selected.

3GPP™ LTE Interleaver

There are 188 standard interleaver sizes from 40 to 6144 bits. To select the internal parameters, set FS low and input the data length into $K[12:0]$. The decoder will automatically select the parameters for that length. Note that the only valid lengths are from 40 to 504 bits that are a multiple of 8, 512 to 1008 bits that are a multiple of 16, 1024 to 2016 bits that are multiple of 32, and 2048 to 6144 bits that are a multiple of 64. Other interleaver lengths will cause incorrect operation.

To input external interleaver parameters, set FS high. Any length from 40 to 6144 bits can be input, provided that K is a multiple of 8. Parameter $F1[8:1]$ is equal to f_1 divided by two. That is, the least significant bit of f_1 is deleted since it is always equal to one due to f_1 being odd. Similarly, parameter $F2[9:1]$ is equal to f_2 divided by two since f_2 is always even. For correct operation, $f_1 < 512$, $f_2 < 1024$, and $f_1 + f_2 < 1024$.

Turbo Decoder Operation

Internal to the turbo decoder are two input and two output RAMs. The received data ready RR signal when high indicates when the decoder can accept data. When low, this indicates the two input buffers are full with their specified data lengths.

To limit input memory complexity, the maximum data length is 20730 for rate 1/2, 12282 for rate 1/3 and 1/4, and 8186 for rate 1/5.

If RR is high, input data can be serially input one 6-bit symbol at a time. The received data start RS signal is used to initialise the internal counters and enable counting. The received data enable RE must be high once and only once for each valid data that is input. The values of K[14:0], N[1:0], NI[7:0], S3G and CRC are internally stored when RS goes high. This data can be input in any sequence, e.g. $K = 40, 20730, 186$, etc. Since RS initialises the internal counters, valid data can only be input after RS goes high.

When all the input data for a specified K[14:0], N[1:0], NI[7:0], S3G and CRC have been input, RR may stay high or go low. If the other input RAM is available, then RR will stay high, indicating that the next block may be input. If both input RAMs are full with their respective block lengths, then RR will go low. RR will not go high again until the turbo decoder has finished decoding the data in one of the input RAMs. Note that RS is ignored while RR is low, preventing data from being written to an input RAM.

The input SYNC is used to select the clock used for the input data signals. When low, the system clock CLK is used. This assumes R[5:0], RS and RE are synchronous to CLK. When SYNC is high, the received data clock RCLK is used. This assumes R[5:0], RS and RE are synchronous to RCLK.

In order to synchronise the received data signals with CLK, internal flip-flops synchronous with CLK are used to detect the rising edge of RCLK. This signal is then used to synchronise the received data with CLK, as well as ensuring that RE is high only once for each received sample. In order for this circuit to correctly function, both the low and high period of RCLK must be greater than the period of CLK. For an even mark to space ratio for RCLK, this implies the frequency of CLK must be more than twice that of RCLK. All other inputs and outputs are synchronous with CLK.

Tables 5 and 6 illustrates the sequence for the main data, the first tail, and the second tail, for 3GPP™ and 3GPP2, respectively. The entries for the table indicate which encoded data output is selected, X, Y1 and Y2 for the first encoder and X', Y1' and Y2' for the second encoder. The code polynomials are $g^0(D) = 1+D^2+D^3$ (13 in octal), $g^1(D) = 1+D+D^3$ (15) and $g^2(D) = 1+D+D^2+D^3$ (17). For rate 1/2 and 1/4 the main data is punctured, which is why two entries are shown.

Table 5: Input data format (3GPP™ LTE)

Rate	Main	Tail 1	Tail 2
1/2	X X	X	X'
	Y1 Y1'	Y1	Y1'
1/3	X	X	X'
	Y1	Y1	Y1'
1/4	X X	X	X'
	Y1 Y1	Y1	Y1'
	Y2 Y2'	0	0
1/5	X	X	X'
	Y1	Y1	Y1'
	Y2	Y2	Y2'
	Y1'	0	0
	Y2'	0	0

For example, for rate 1/2, the main data is input in the order X, Y1, X, Y1', X, Y1, etc.

The decoded block is output from one of the output RAMs after a block has been decoded. The signal XDR goes high for KD[14:0] CLK or RCLK cycles (for SYNC low or high, respectively) while the block is output. The block is output in sequential order with address XDA[14:0].

When SYNC is low, the output signals will be delayed by one to two CLK cycles after the rising edge of RCLK.

Table 6: Input data format (3GPP2 1xEV-DO)

Rate	Main	Tail 1	Tail 2
1/2	X X	X	X'
	Y1 Y1'	Y1	Y1'
1/3	X	X	X'
	Y1	X	X'
1/4	X X	X	X'
	Y1 Y1	X	X'
	Y2 Y2'	Y1	Y1'
1/5	X	X	X'
	Y1	X	X'
	Y2	Y1	Y1'
	Y1'	Y2	Y2'
	Y2'	Y2	Y2'

The signal ERR is the cycle redundancy check of the turbo decoded block. It will be valid for at

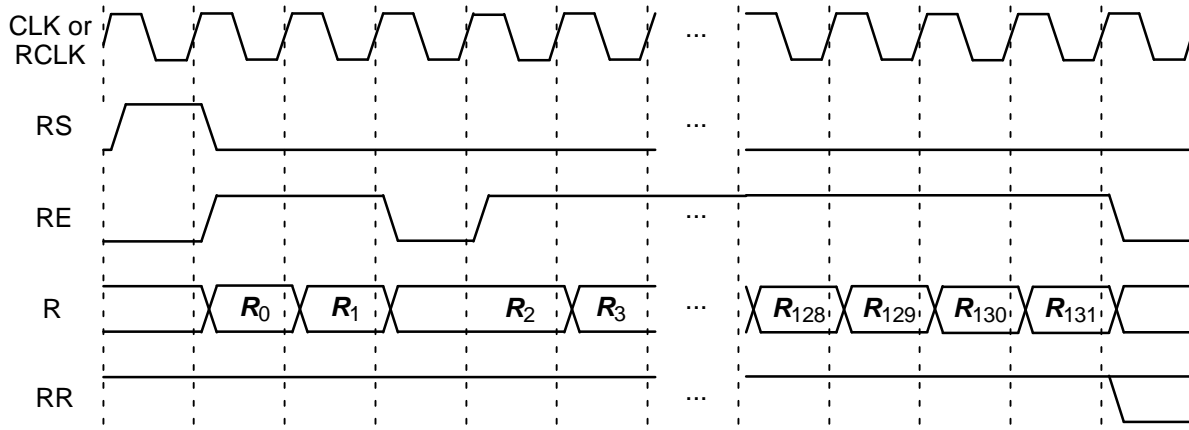


Figure 4: Turbo Decoder Input Timing ($K = 40$, LTE).

least one CLK or RCLK cycle (for SYNC low or high, respectively) after XDR goes low, depending on whether the CRC for the data block is satisfied. ERR will be low if the CRC is satisfied and high if the CRC is not satisfied, indicating there are errors in the decoded block.

The input CRC selects the parity check polynomial. When low, the 16-bit CRC with polynomial $1+D^5+D^{12}+D^{16}$ is selected. When CRC is high, the 24-bit CRC with polynomial $1+D+D^5+D^6+D^{23}+D^{24}$ is selected.

Figure 5 illustrates the decoder timing. Ignoring the number of clock cycles required to input and output the data, the maximum number of clock cycles for decoding is $(N+1)(K+L+1)+1$.

The early stopping algorithm uses the magnitude of the extrinsic information to determine when to stop. As the decoder iterates, the magnitudes generally increases in value as the decoder becomes more confident in its decision. By comparing the smallest magnitude of a block with threshold ZTH, we can decide when to stop. If the

smallest magnitude is greater than ZTH, i.e., not equal or less than ZTH, the decoder will stop iterating if early stopping has been enabled.

Since the last half iteration is used to store the decoded data into the interleaver memory, the decoder performs an extra half iteration once the threshold has been exceeded.

Increasing ZTH will increase the average number of iterations and decrease the BER. Decreasing ZTH will decrease the average number of iterations and increase the BER. In general, higher values of SNR will decrease the number of iterations. A value of ZTH = 23 was found to give a good trade off between the average number of iterations and BER performance.

For high SNR operation early stopping can lead to significantly reduced power consumption, since most blocks will be decoded in one or two iterations.

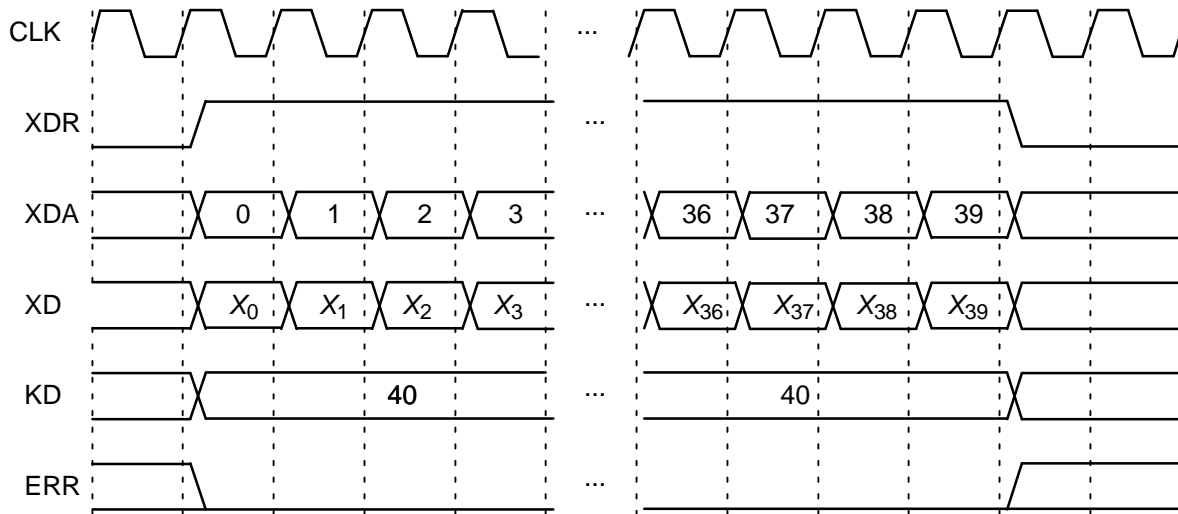


Figure 5: Turbo Decoder Output Timing ($K = 40$).

Simulation Software

Free software for simulating the PCD03VH turbo decoder in additive white Gaussian noise (AWGN) or with external data is available by sending an email to info@sworld.com.au with "pcd03vsim request" in the subject header. The software uses an exact functional simulation of the PCD03VH turbo decoder, including all quantisation and limiting effects.

After unzipping `pcd03vsim.zip`, there should be `pcd03vsim.exe` and `code.txt`. The file `code.txt` contains the parameters for running `pcd03vsim`. These parameters are

<code>m</code>	Constituent code (CC) memory (2 to 4)
<code>nt</code>	Number of turbo code outputs (2 to 5)
<code>g0</code>	Divisor polynomial of CC in octal notation
<code>g1</code>	1st numerator polynomial of CC
<code>g2</code>	2nd numerator polynomial of CC
<code>EbNomIn</code>	Minimum E_b/N_0 (in dB)
<code>EbNomax</code>	Maximum E_b/N_0 (in dB)
<code>EbNoInC</code>	E_b/N_0 increment (in dB)
<code>optC</code>	Input scaling parameter (normally 0.65)
<code>ferrmax</code>	Number of frame errors to count
<code>Pfmin</code>	Minimum frame error rate (FER)
<code>Pbmin</code>	Minimum bit error rate (BER)
<code>NI</code>	Number of half iterations–1 (0 to 255)
<code>SLD</code>	MAP decoder delay select (0 or 1)
<code>LIMZ</code>	Extrinsic information limit (1 to 127)
<code>SCLZ</code>	Extrinsic information scale (1 to 32)
<code>M</code>	Stopping mode (0 to 3)
<code>ZTH</code>	Extrinsic info. threshold (0 to 127)
<code>S3G</code>	Standard select (0 to 3)
<code>CS</code>	Use external 3GPP2 interleaver parameters (0 or 1)
<code>K</code>	Block length (40 to 5114 for 3GPP™ or 17 to 21504 for 3GPP2)
<code>q</code>	Number of quantisation bits (3 to 6)
<code>LOGMAP</code>	Log–MAP decoding (MODE0, 0 or 1)
<code>C4PIN</code>	Use five–bit C (MODE1, 0 or 1)
<code>state</code>	State file (0 to 2)
<code>s1</code>	Seed 1 (1 to 2147483562)
<code>s2</code>	Seed 2 (1 to 2147483398)
<code>fs</code>	LTE interleaver parameter select (0 or 1)
<code>f1</code>	LTE interleaver parameter 1 (1 to 511)
<code>f2</code>	LTE interleaver parameter 2 (0 to 1022)
<code>read_x</code>	Use external information data (y or n)
<code>read_r</code>	Use external received data (y or n)
<code>out_dir</code>	Output directory
<code>in_dir</code>	Input directory
<code>out_screen</code>	Output data to screen (y or n)
<code>C</code>	C (0 to 17)

Note that `g0`, `g1` and `g2` are given in octal notation, e.g., `g0 = 13` $\equiv 1011_2 \equiv 1+D^2+D^3$. For the 3GPP™ standard, `m = 3`, `nt = 3`, `g0 = 13` and `g1`

= 15 (`g2` is not used). The nominal turbo code rate is $1/nt$.

The parameter `optC` is used to determine the "optimum" values of A and C . The "optimum" value of A is

$$A = \frac{\text{optC}(2^{q-1} - 1)}{\text{mag}(\sigma)} \quad (4)$$

where σ^2 is the normalised noise variance given by (2) and $\text{mag}(\sigma)$ is the normalising magnitude resulting from an auto–gain control (AGC) circuit. We have

$$\text{mag}(\sigma) = \sigma \sqrt{\frac{2}{\pi}} \exp\left(\frac{-1}{2\sigma^2}\right) + 1 - 2Q\left(\frac{1}{\sigma}\right) \quad (5)$$

where $Q(x)$ is the error function given by

$$Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{t^2}{2}\right) dt. \quad (6)$$

Although $\text{mag}(\sigma)$ is a complicated function, for high signal to ratio (SNR), $\text{mag}(\sigma) \approx 1$. For low SNR, $\text{mag}(\sigma) \approx \sigma \sqrt{2/\pi} \approx 0.798\sigma$. That is, an AGC circuit for high SNR has an amplitude close to the real amplitude of the received signal. At lower SNR, the noise increases the estimated amplitude, since an AGC circuit averages the received signal amplitude.

For the "optimum" A , we round the value of C given by (1) to the nearest integer. If `LOGMAP = MODE0 = 0` then C is forced to 0. If `LOGMAP = 1` and `C4PIN = MODE1 = 0`, C is limited to a maximum value of 9. If `LOGMAP = 1` and `C4PIN = 1`, C is limited to a maximum value of 17.

The simulation will increase E_b/N_0 (in dB) in `EbNoInC` increments from `EbNomIn` until `EbNomax` is reached or the frame error rate (FER) is below or equal to `Pfmin` or the bit error rate (BER) is below or equal to `Pbmin`. Each simulation point continues until the number of frame errors is equal to `ferrmax`. If `ferrmax = 0`, then only one frame is simulated.

When the simulation is finished the output is given in, for example, file `k600.dat`, where $K = 600$. Only frames that are in error are stored in the output file. The first line gives the E_b/N_0 (`Eb/No`), the number of frames (`num`), the number of bit errors in the frame (`err`), the total number of bit errors (`berr`), the total number of frame errors (`ferr`), the average number of iterations (`na`), and the average BER (`Pb`) and the average FER (`Pf`). Following this, the number of iterations, `na`, `berr`, `ferr`, `Pb`, and `Pf` are given for each half iteration.

The following file was used to give the simulation results shown in Figure 6. Auto–stopping was used with up to 10 iterations. When iterating is

stopped early, the `nasum` ($2 \times \text{num} \times \text{na}$), `berr` and `ferr` results at stopping are copied for each half iteration to the maximum iteration number. Thus, the $I = 10$ result is the performance one would measure with auto-stopping and $NI = 19$. The $I = 2.5$ and 5 curves show the performance at 2.5 and 5 iterations with early stopping and $NI = 4$ and 9, respectively. Figure 7 shows the average number of iterations with E_b/N_0 . Figure 8 shows the performance with $NI = 19$, $n = 2$ and 3, and $K = 200$, 500 and 1000.

```
{m nt g0 g1 g2}
 3 3 13 15 17
{EbNomin EbNomax EbNoinc optC ferrmax
Pfmin Pbmin}
 0.00 1.50 0.25 0.65 64
1e-99 1e-99
{NI SLD LIMZ SCLZ M ZTH S3G CS}
 19 1 96 32 3 23 0 0
{K q LOGMAP C4PIN}
 600 6 1 1
{state s1 s2 fs f1 f2}
 0 12345 67890 0 3 10
{read_x read_r out_dir in_dir
out_screen C}
 n n output input
y 12
```

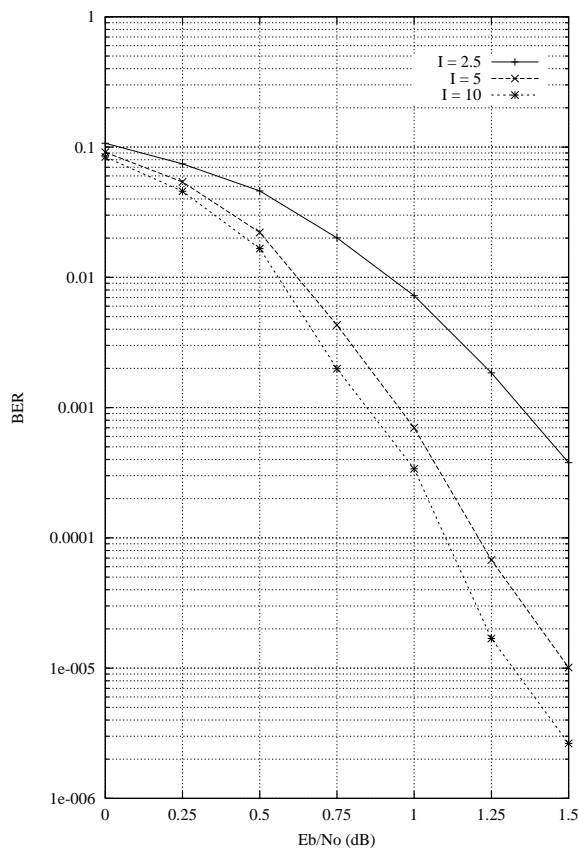


Figure 6: Performance with block size 600, 3GPP™ and auto-stopping (ZTH = 23).

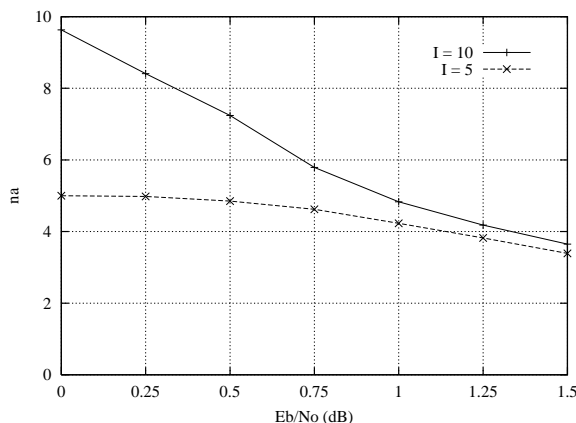


Figure 7: Average number of iterations with block size 600 and auto-stopping (ZTH = 23).

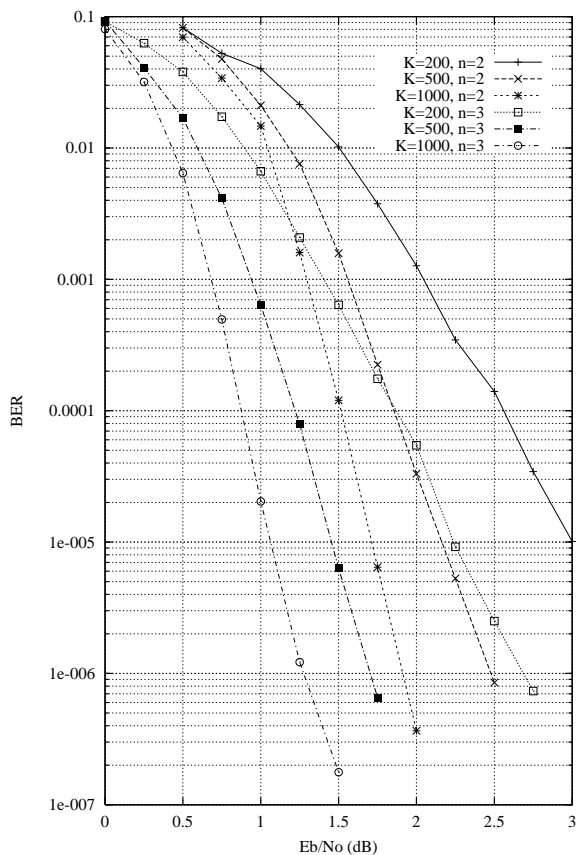


Figure 8: Performance with auto-stopping and 10 iterations.

The `state` input can be used to continue the simulation after the simulation has been stopped, e.g., by the program being closed or your computer crashing. For normal simulations, `state = 0`. While the program is running, the simulation state is alternatively written into `State1.dat` and `State2.dat`. Two state files are used in case the program stops while writing data into one file. To continue the simulation after the program is stopped follow these instructions:

- 1) Copy the state files State1.dat and State2.dat. This ensures you can restart the program if a mistake is made in configuring code.txt.
- 2) Examine the state files and choose one that isn't corrupted.
- 3) Change the state parameter to 1 if State1.dat is used or 2 if State2.dat is used.
- 4) Restart the simulation. The output will be appended to the existing k(K).dat file.
- 5) After the simulation has been completed, make sure that state is changed back to 0.

The software can also be used to encode and decode external data. To encode a block $X_{(K)}.dat$ in the directory given by `in_dir`, set `read_x` to `y`, e.g., $X_{600}.dat$ in directory `input` (each line contains one bit of data). The encoded stream $Y_{(K)}.dat$ will be output to the directory given by `out_dir`, e.g., $Y_{600}.dat$ to directory `output`.

To decode data, place the received block of data in file $R_{(K)}.dat$ in directory `in_dir` and set `read_r` to `y`. The decoded data is output to $XD_{(K)}.dat$ in directory `out_dir`. $R_{(K)}.dat$ has in each line $R[i,j]$, $i = 0$ to $nt-1$ from $j = 0$ to $K+2m-1$, e.g., for $nt = 3$ the first three lines could be

```
-31 1 -25
-31 12 9
11 31 31
```

The input data is of the form

$$R[i,j] = A*(1-2*Y[i,j]+N[i,j])$$

where A is the signal amplitude, $Y[i,j]$ is the coded bit, and $N[i,j]$ is white Gaussian noise with zero mean and normalised variance σ^2 . The magnitude of $R[i,j]$ should be rounded to the nearest integer and be no greater than $2^{q-1}-1$. If `read_r = y`, then C is externally input via `c`.

For 3GPP2, external interleaver parameters can be used. Set `cs = 1` and place the 32 interleaver parameters, one parameter per line, in file `c.dat` in directory `in_dir`. The full value should be input, and not the value with the lsb removed.

Ordering Information

SW-PCD03VH-SOS (SignOnce Site License)
 SW-PCD03VH-SOP (SignOnce Project License)
 SW-PCD03VH-VHD (VHDL ASIC License)
 SW-PCD03VH-UNI-n (University License)

All licenses include EDIF and VHDL cores. The VHDL cores can only be used for simulation in the SignOnce and University licenses. The Uni-

versity license is only available to tertiary educational institutions such as universities and colleges and is limited to n instantiations of the core. The SignOnce and ASIC licenses allows unlimited instantiations.

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] Third Generation Partnership Project (3GPP), "Evolved universal terrestrial radio access (E-UTRA); Multiplexing and channel coding," 3GPP TS 36.212 V8.1.0 Release 8, Nov. 2007.
- [2] Third Generation Partnership Project 2 (3GPP2), "cdma2000 high rate packet data air interface specification, Release B," 3GPP2 C.S0024-B Version 2.0, Mar. 2007.
- [3] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284-287, Mar. 1974.
- [4] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," *ICC'95*, Seattle, WA, USA, pp. 1009-1013, June 1995.
- [5] M. C. Reed and J. A. Asenstorfer, "A novel variance estimator for turbo-code decoding," *Int. Conf. on Telecommun.*, Melbourne, Australia, pp. 173-178, Apr. 1997.

Small World Communications does not assume any liability arising out of the application or use of any product described or shown herein; nor does it convey any license under its copyrights or any rights of others. *Small World Communications* reserves the right to make changes, at any time, in order to improve performance, function or design and to supply the best product possible. *Small World Communications* will not assume responsibility for the use of any circuitry described herein. *Small World Communications* does not represent that devices shown or products described herein are free from patent infringement or from any other third party right. *Small World Communications* assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made. *Small World Communications* will not assume any liability for the accuracy or correctness of any engineer-

ing or software support or assistance provided to a user.

© 2000–2011 *Small World Communications*. All Rights Reserved. Xilinx, Spartan and Virtex are registered trademarks of Xilinx, Inc. All XC–prefix product designations are trademarks of Xilinx, Inc. 3GPP is a trademark of ETSI. All other trademarks and registered trademarks are the property of their respective owners.

Supply of this IP core does not convey a license nor imply any right to use turbo code patents owned by France Telecom, GET or TDF. Please contact France Telecom for information about turbo codes licensing program at the following address: France Telecom R&D – VAT/Turbo-codes, 38 rue du Général Leclerc, 92794 Issy Moulineaux Cedex 9, France.

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

Revision History

v0.20 1 September 2008. Preliminary product specification.

v1.00 15 September 2008. First official release.

v1.02 14 April 2009. Updated Virtex 4 and Virtex 5 complexity. Corrected Virtex–4 part number.

v1.03 3 March 2010. Added equation for NI in signal descriptions. Corrected description for soft decision inputs that are less than six bits.

v1.04 30 May 2011. Changed BER simulation program name from pcd03sim to pcd03vsim. Corrected fading channel information. Added revision history.