White Paper

Area Optimized Soft Decision Viterbi Decoder Functions

Introduction

The Altera® area optimized, soft decision Viterbi decoder HammerCores are optimized for APEX™20K, FLEX®10K and FLEX 6000 devices. You can parameterize the devices by implementing any number of standard decoders or you can quickly realize a custom application.

The functions use a memory and logic based solution for area efficient implementations. Five types of decoder functions are included in the library:

- VSAA- serial processing decoder
- VSBERAA- VSAA function with built-in bit error rate (BER) measurement
- VSPAA- serial/parallel processing decoder
- VSPBERAA- VSPAA function with built-in BER measurement
- VSPDLAB- serial/parallel processing decoder with dual code support

Processing of the Viterbi algorithm is serial for the VSAA and VSBERAA decoders. The VSPAA, VSPBERAA, and VSPDLAB decoders use multiple serial processing elements for higher throughput. Function throughput is largely dependent on constraint length. Table 1 shows a typical expected performance for function and constraint length combinations for -1 speed grade APEX 20KE devices. The throughput is expressed in decoded bit rate out.

Table 1. Function Performance

Constraint Length	Performance	
	Serial Function (Mbps)	Serial/Parallel Function (Mbps)
5	7.8	N/A
7	2	8
9	0.5	2

The decoders also support hard decision decoding when the number of soft bits are set to two. The function includes several utilities to generate test cases and to analyze the results. The results allow you to verify BER performance of the decoders to various channels.

Ports and Parameters

Table 2 shows the parameters for VSSA and VSPAA functions.

Table 2. Parameters

Parameters	Description
N	This is the number of coded bits. For every encoded bit, *n* bits are output. This parameter can range from two to seven.
L	This is the constraint length. The constraint length can vary between five and nine for VSAA, and seven and nine for VSPAA.
SOFTBITS	The number of soft decision bits per symbol. The range is 2 bits and up. When SOFTBITS is set to 2 bits, the decoder acts as a hard decision decoder and allows for erased symbols to be entered as binary '00'.
BMGWIDE	This is the precision of the branch metric accumulation. It can be any value, approximately SOFTBITS + 4 bits or greater. The decoder may flag and error when compiling for certain combinations of N and L, and specify a larger number for BMGWIDE. When the decoder is reset, all metrics are set to zero. Once any metric reaches '100.....', all metrics are normalized by diving their values in half.
V	This parameter is the traceback depth. It is typically set to 5L for unpunctured codes, and up to 15L for highly punctured codes. It must be set to 10 or greater.
GA, GB, GC, GD, GE, GF, GG	A total of N generator polynomials will be required, in decimal form. If N is less than 7, the unused polynomials should be set to zero.

The choice of V and L will affect the number of output bits decoded at the same time. The number of output bits bitsout will be:

ceil(V/($2^{(L-1)}$) for VSAA, and

ceil(V/($2^{(L-3)}$) or VSPAA.

V is restricted in certain ranges, depending on L. V must be:

($x.2^{(L-1)} + 2$) < V < ((x+1).$2^{(L-1)}$) for VSAA, and

($x.2^{(L-3)} + 2$) < V < ((x+1).$2^{(L-3)}$) for VSPAA,

where x is an integer.

Table 3 shows the input signals for VSAA and VSPAA functions.

Table 3. Input Signals

Signal	Description
SYSCLK	This is the main system clock.
RESET	The entire decoder is asynchronously reset when RESET is asserted high, then decoding of a new block can start.
ENABLE	The operation of the decoder is enabled when ENABLE is asserted high. When it is asserted low, all processing stops.
LOAD	This signal latches the RR [ ] bus into the decoder.
RR [(n*SOFTBITS)..1]	This bus takes in one N symbol, each SOFTBITS wide. The first symbol received will occupy the most significant bits of RR [ ], and the last (*n*th) symbol received will occupy the least significant bits of RR [ ]. This can be seen in the test cases generated by the **VVECG.EXE** utility. Erased (de-punctured) symbols are equal to zero. An encoded '1' is negative (i.e. '1XX...') and an encoded '0' is positive (i.e. 'OXX..').

Table 4 shows the output signals for the VSAA and VSPAA functions.

Table 4. Output Signals

Signal	Description
Ready	This signal is asserted when the decoder requires another N symbols on the RR [ ] bus on the next rising edge. If the input symbols are not available, the decoder must be disabled until they are available (using the ENABLE signal).
DECBITS [bitsout..1]	This bus (it may only be one bit wide) contains output bits when OUTVALID is asserted. The value of bitsout is explained in the V parameter.
OUTVALID	The signal is asserted high for one clock cycle whenever there is a valid output on the DECBITS [ ] bus.
NORMALIZE	This signal is asserted high for one clock cycle whenever the branch metrics are normalized.

VSBERAA & VSPBERAA Functions

The VSBERAA and VSPBERAA functions are supersets of the VSAA function, respectively, with BER measurement added. The function parameters are identical (see Table 2), but the allowable ranges differ (see Table 5). There are also additional ports. The BER measurement is estimated using a re-encode (of the decoded data) and compare method.

Table 5. VSBERAA & VSPBERAA Function Parameters Note (1)

Signal	Description
V	The traceback parameter has additional restrictions for the VSBERAA and VSPBERAA functions. V must be evenly divisible by bitsout.

Note:
(1) When DECBITS is greater than one, the number of decoded symbol sets is used to count the measurement period, rather than the number of decoded symbols, i.e. the period will actually be DECBITS times as long as specified.

Table 6 shows input signals for VSBERAA and VSPBERAA functions.

Table 6. VSBERAA & VSPBERAA Function Input Signals

Signal	Description
PERIOD [24..1]	This bus specifies the number of decoded symbols over which the BER measurement is made. After the specified number of symbols is reached, the BERERR [16..1] output bus is latched with the number of errors estimated during the previous measurement period, and the NUMERR [16..1] bus is reset to zero. Then, a new measurement period begins.

Table 7 shows output signals for VSBERAA and VSPBERAA functions.

Table 7. VSBERAA & VSPBERAA Function Output Signals

Signal	Description
NUMERR [16..1]	This bus contains the number of errors detected during the current measurement period. It is updated each time that an error is detected, making it possible to see the location of individual errors. It is reset at the end of each measurement period.

BERERR [16..1]	This bus contains the number of errors detected during the previous measurement period. It is updated at the end of each measurement period.

VSPDLAB Function

The VSPDLAB is a serial/parallel function that supports two codewords at a time. The selection between codewords is made with an input signal. BERT is not available for this function. Table 8 shows the VSPDLAB function parameters.

Table 8. VSPDLAB Function Parameters

Signal	Description
N_ONE	This is the number of coded bits for the first codeword. For every bit to be encoded, N_ONE bits are output. This parameter can range from two to seven.
N_TWO	This is the number of coded bits for the first codeword. For every bit to be encoded, N_TWO bits are output. This parameter can range from two to seven. If N_ONE and N_TWO are different, N_TWO must be the larger value.
L	This is the constraint length. The constraint length can vary between seven and nine. Only one constraint length is supported at any one time.
SOFTBITS	This is the number of soft decision bits per symbol. The range is 2 bits and up. When SOFTBITS is set to 2 bits, the decoder acts as a hard decision decoder and allows for erased symbols to be entered as binary '00'. Only one value of SOFTBITS is supported at any one time.
BMGWIDE	This is the precision of the branch metric accumulation. It can be any value, approximately SOFTBITS + 4 bits or greater. The decoder may flag an error when compiling for certain combinations of N and L and specify a larger number for BMGWIDE. When the decoder is reset, all metrics are set to zero. Once any metric reaches '100...', all metrics are normalized by dividing their values in half. Only one value of BMGWIDE is supported at any one time.
V	This parameter is the traceback depth. It is typically set to 5L for unpunctured codes, and up to 15L for highly punctured codes. It must be set to 10 or greater. Only one value of V is supported at any one time.
GA_ONE, GB_ONE, GC_ONE, GD_ONE, GE_ONE, GF_ONE, GG_ONE	For the first codeword, a total of N_ONE generator polynomials will be required in decimal form. If N_ONE is less than seven, the unused polynomials should be set to zero.
GA_TWO, GB_TWO, GC_TWO, GD_TWO, GE_TWO, GF_TWO, GG_TWO	For the second codeword, a total of N_TWO generator polynomials will be required in decimal form. If N_TWO is less than seven, the unused polynomials should be set to zero.

The choice of V and L affects the number of output bits decoded at the same time. The number of output bits will be:

$$\text{ceil}(V/(2^{(L-3)})$$

V is restricted in certain ranges, depending on L. V must be:

$$(x.2^{(L-3)} + 2) < V < ((x+1).2^{(L-3)})$$

where x is an integer.

Many of the ports for VSPDLAB are identical to those for VSPAA (see Tables 3 and 4). Only the ports that are different are explained in Table 9.

Table 9. VSPDLAB Function Input Signals

Signal	Description
SELCODE	When low, the first codeword is selected. When high, the second codeword is selected.
RR [(n_two*softbits)..1]	This bus takes in one N_TWO symbols, each SOFTBITS wide. The first symbol received will occupy the most significant bits of RR [], and the last (N_TWOth) symbol received will occupy the least significant bits of RR []. When N_ONE and N_TWO differ, the symbols for the first codeword will occupy the lower (N_ONE*SOFTBITS) bits of RR [].

Soft Symbol Inputs

Table 10 shows an example of the soft symbol input representation for SOFTBITS= 4.

Table 10. Soft Symbol Input Representation

Soft Symbol	Meaning
0111	Strongest '0'
0110	
0101	
0100	
0011	
0010	
0001	Weakest '0'
0000	Erased Symbol
1111	Weakest '1'
1110	
1101	
1100	
1011	
1010	
1001	Strongest '1'
1000	Stronger '1' (input normally clipped to 1001 for maximum '1')

Recommended Generators for Code Selection

Tables 11 and 12 list maximum distance codes when parameter N is two and three, with constraint lengths from L= 5 to L= 9. The generators are given in both octal and decimal values. The function parameters are entered in decimal form.

Table 11. Maximum Distance Codes for N= 2

Constraint Length	Generators in Octal		Generators in Octal	
	GA	GB	GA	GB
5	23	35	19	29
6	53	75	43	61

7	133	171	91	121
8	247	371	167	249
9	561	753	369	491

Table 12. Maximum Codes for N= 3

Constraint Length	Generators in Octal			Generators in Octal		
	GA	GB	GC	GA	GB	GC
5	25	33	37	21	27	31
6	47	53	75	39	43	61
7	133	145	175	91	101	125
8	255	331	367	149	219	247
9	557	663	711	367	435	457

Puncturing Scheme

All punctured codes shown are based on a mother code rate of 1/2. All of the decoders require you to de-puncture the received data stream external to the decoder and input the data into the decoder `N` symbols at a time. The puncturing rates supported are 0 (unpunctured), 2/3, 3/4, 4/5, 5/6, 6/7, and 7/8. Table 13 shows the puncturing scheme rates.

The test cases can be setup for these punctured codes by the **VVECG.EXE** utility. The test cases have the input streams de-punctured (by zero insertion).

Table 13. Puncturing Scheme Used by Viterbi Functions

Rate	Puncturing Scheme							
	Bit	Multiplier						
2/3	CA	1	0					
	CB	1	1					
3/4	CA	1	0	1				
	CB	1	1	0				
4/5	CA	1	0	0	0			
	CB	1	1	1	1			
5/6	CA	1	0	1	0	1		
	CB	1	1	0	1	0		
6/7	CA	1	0	0	1	0	1	
	CB	1	1	1	0	1	0	
7/8	CA	1	0	0	0	1	0	1
	CB	1	1	1	1	0	1	0

CA refers to the most significant (first transmitted bit, first received symbol) and CB refers to the least significant (last transmitted bit, last received symbol).

Other Punctured Codes

Other punctured codes, either with different puncturing matrixes for codes based on a mother rate of 1/2, or punctured codes based on mother rates of 1/3 or 1/4 can be implemented by you. If the state machine controlling the de-puncturing requires to halt the decoder while zeros are inserted into the symbol stream, `ENABLE` on the decoder can be de-asserted.

VSAA, VSBERAA Function Compilation

The VSAA and VSBERAA decoders take in N symbols per load and return DECBITS decoded bits per clock. The output bit rate of the functions is the system clock rate, divided by $2^{(L-1)}$.

If a punctured code is being used, you must de-puncture the received bits externally to the function. Because there are a number of clocks required for each N symbol inputs, the de-puncturing can be done between inputs. Even if the code is unpunctured, you must still combine N symbols into a parallel symbol vector and present the symbol vector to the decoder with every decoder clock.

When testing the VSAA or VSBERAA decoder functions with a punctured code, erased symbols are entered as zero.

Setting Compilation Options

The functions should be compiled with **Global Logic Synthesis** set to fast.

Example Decoder Design-VSAA

The following steps are used to design a rate1/2 constraint length 7, Viterbi decoder.

The parameters are:

N= 2

SOFTBITS= 3

L= 7

GA= 91

GB= 121

GC= 0

GD= 0

GE= 0

GF= 0

GG= 0

BMGWIDE= 14

V= 35

The Altera MAX+PLUS® II Software was used to compile the top level file, **VSAA.TDF**. The target device was an Altera EPF10K30ETC144-1 device. The design required 736 LCs and 6 EABs. At a system clock rate of 122 MHz, the throughput was 1.9 Mbps.

The Altera Quartus™ software was also used to compile the design targeted for an EP20K60E device. The design required 635 LCs, and 6 ESBs. At a system clock rate of 135 MHz, the throughput was 2.1 Mbps.

Example Decoder Design-VSAA

The following steps are used to design at a rate 1/2 constraint length 9, Viterbi decoder.

The parameters are:

N= 2

SOFTBITS= 3

L= 9

GA= 369

GB= 491

GC= 0

GD= 0

GE= 0

GF= 0

GG= 0

BMGWIDE= 15

V= 50

The MAX + PLUS II software was used to compile the top level file, **VSAA.TDF**. The target device was an EPF10K200SQC240-1. This implementation would also be more efficient and faster than a EP20K100E device. The design required 1758 LCs and 21 ESBs. At a system clock rate of 74 MHz, the throughput was 289 Kbps.

The Quartus software was also used to compile the design targeted for an EP20K100E device. The design required 1461 LCs and 21ESBs. At a system clock rate of 105 MHz, the throughput was 410 Kbps.

Example Decoder Design-VSBERAA

The following steps are used to design a rate 1/3 constraint length 5, Viterbi decoder with BERT functionality.

The parameters are:

N= 3

SOFTBITS= 4

L= 5

GA= 21

GB= 27

GC= 31

GD= 0

GE= 0

GF= 0

GG= 0

BMGWIDE= 15

V= 360

The MAX + PLUS II software was used to compile the top level file, **VSBERAA.TDF**. The target device was a EPF10K30ETC144-1. The design required 735 LCs and 4 ESBs. At a system clock rate of 100 MHz, the throughput was 6.25 Mbps.

The Quartus software was also used to compile the design with an EP20K60E target device. The design required 698 LCs and 4 ESBs. At a system clock rate of 116 MHz, the throughput was 7.25 Kbps.

VSPAA, VSPBERAA, & VSPDLAB Function Compilation

The VSPAA, VSPBERAA, and VSPDLAB decoders take in N symbols per load and returns DECBITS decoded bits per clock. The output bit rate of the function is the system clock rate, divided by $2^{(L-3)}$.

If a punctured code is being used, you must de-puncture the received bits externally to the function. Because there are a number of clocks required for each N symbol inputs, the de-puncturing can be done between inputs. Even if the code is unpunctured, you must still combine N symbols into a parallel symbol vector (Ports and Parameters), and present the symbol vector to the decoder with every decoder clock.

When testing the decoder functions with a punctured code, erased symbols are entered as zero.

Setting Compilation Options

The functions should be compiled with the **Global Logic Synthesis** option set to fast.

Example Decoder Design- VSPAA

The following steps are used to design rate 1/2 constraint length 7, Viterbi decoder.

The parameters are:

N= 2

SOFTBITS= 3

L= 7

GA= 91

GB= 121

GC= 0

GD= 0

GE= 0

GF= 0

GG= 0

BMGWIDE= 14

V= 100

The MAX+PLUS II software was used to compile the top level file, **VSPAA.TDF**. The target device was an EPF10K50ETC144-1. The design required 1475 LCs and 10 ESBs. At a system clock rate of 91 MHz, the throughput was 5.7 Mbps.

The Quartus software was also used to compile the design with an EP20K100E target device. The design required 1351 LCs and 10 ESBs. At a system clock rate of 110 MHz, the throughput was 6.9 Mbps.

Example Decoder Design- VSPBERAA

The following steps are used to design a rate 1/2 constraint length 7, Viterbi decoder.

The parameters are:

N= 2

SOFTBITS= 3

L= 7

GA= 91

GB= 121

GC= 0

GD= 0

GE= 0

GF= 0

GG= 0

BMGWIDE= 14

V= 105

The MAX+PLUS II software was used to compile the top level file, **VSPBERAA.TDF**. The target device was an EPF10K50ETC144-1. The design required 1890 LCs and 13 ESBs. At a system clock rate of 82 MHz, the throughput was 5.125 Mbps.

The Quartus software was also used to compile the design with an EP20K100E device. The design required 1639 LCs and 13 ESBs. At a system clock rate of 109 MHz, the throughput was 6.9 Mbps.

Example Decoder Design- VSPDLAB

The following steps are used to design a constraint length 7 decoder, with both rate 1/2 and 1/3 codes.

The parameters are:

N_ONE= 2

N_TWO= 3

SOFTBITS= 3

L= 7

GA_ONE= 91

GB_ONE= 121

GC_ONE= 0

GD_ONE= 0

GE_ONE= 0

GF_ONE= 0

GG_ONE= 0

GA_TWO= 91

GB_TWO= 101

GC_TWO= 125

GD_TWO= 0

GE_TWO= 0

GF_TWO= 0

GG_TWO= 0

BMGWIDE= 15

V= 50

The MAX+PLUS II software was used to compile the top level file, **VSPDLAB.TDF**. The target device was an EPF10K100EQC208-1. This implementation would also be more efficient and faster in an EPF20K100E device.

The Quartus software was also used to compile the design with an EP20K100E as the target device. The design required 1617 LCs and 12 ESBs. At a system clock rate of 116 MHz, the throughput was 7.3 Mbps.

Generating Test Vectors for Function Testing

A test vector generation program, **VVECG.EXE**, is used to create test cases for all five decoder functions. Also, test cases with the same symbol sequences are generated for the High Performance HammerCores by Altera Viterbi decoder functions.

The **VVECG** utility uses the following inputs:

- `BITS`: The number of bits to be coded for the test case.
- `Eb/No`: The SNR for the AWGN channel used in the test case.
- `SOFTBITS`: The parameter `SOFTBITS` of the decoder.
- `N`: The parameter `N` of the decoder.
- `L`: The parameter `L` of the decoder.
- `V`: The parameter `V` of the decoder.
- Rate: The punctured code rate of the channel. The test case contain a de-punctured stream of received symbols. This input is given as a value rather than a string. Valid inputs are: 0, 23, 34, 45, 56, 67, and 78.
- Generator polynomials, in decimal.
- `-r` (optional): this switch is used to specify the value of `N_TWO` for the VSPDLAB decoder when the rate of the two codewords is different. Example use: `-r 4`.

Running the program without a complete set of inputs will display the required inputs. When **VVECG** is run with a complete parameter list, it will output the following results:

- List of generator polynomials, in decimal.
- Actual BER for the test case. As the error locations are randomly generated, the actual BER may differ slightly from the expected BER for a given SNR. The number of bit errors will also be displayed.

The BER rate may differ from what you are expecting because there are many different modulation formats which have differing, uncoded BER rates. The BER rates are output so that you can verify the decoder performance to BER in.

The program will generate the following vector (for the MAX+PLUS II software simulator) files:

- **VSAA.VEC**- a MAX+PLUS II vector for testing the VSAA decoder.
- **VSBERAA.VEC**-a MAX+PLUS II vector file for testing the VSBERAA decoder.
- **VSPAA.VEC**- a MAX+PLUS II vector file for testing the VSPAA decoder.
- **VSPBERAA.VEC**- a MAX+PLUS II vector file for testing the VSPBERAA decoder.
- **VSPDLAB.VEC**- a MAX+PLUS II vector file for testing the VSPDLAB decoder.
- **VITTOPA.VEC**- a MAX+PLUS II vector file for testing the VITTOPA (High Performance) decoder.
- **VITTOPB.VEC**- a MAX+PLUS II vector file for testing the VITTOPB (High Performance) decoder.

The following text files will also be generated to analyze the simulation results.

- **TRANSBIT.TXT**- randomly generated bits coded in the test case.
- **A_TXSYM.TXT**- encoded symbol vectors for the test case, the serial decoders, and the VITTOPA decoders. The transmitted symbols are '1' and '0'.
- **A_RXSYM.TXT**- encoded symbol vectors with the AWGN channel added for the serial and VITTOPA decoders. Each symbol is of SOFTBITS precision.
- **A_ERRLOC.TXT**- locations by symbol vector of the received errors. More than one error can be indicated by symbol vector, as N symbols are received per clock.
- **B_TXSYM.TXT**- encoded symbols for the VITTOPB decoder test case.
- **B_RXSYM.TXT**- encoded symbols with the AWGN channel added for the VITTOPB decoder.
- **B_ERRLOC.TXT**- locations by symbol, of received errors.

Running the Test Case

To run a test case, you must first load the appropriate Vector File (**.vec**) for the decoder to be tested into the Altera simulator.

To simulate, open the MAX+PLUS II software simulator window and select **Inputs/Outputs** (File menu). Select the appropriate **.VEC** file that was created in the folder where **VVECG.EXE** is located. The simulator converts the **VEC** file into a simulator channel file (**.scf)** file.

Simulate. When simulation is complete, open the **SCF** file. Select **Create Table File** (File menu) to generate a text output of the simulation. This text output can be read by **VSTBLAB.EXE** (explained in Analyzing Test Results).

For the VSBERAA and VSPBERAA decoders simulation, the PERIOD[] bus should be set to a non-zero value.

For the VSPDLAB decoder simulation, SELCODE should be set high if the second codeword is tested.

Analyzing Test Results

All functions can be analyzed by the **VSTBLAB.EXE** utility.

The **VSTBLAB** program requires the following parameters:

- Bits- the number of bits to compare.
- Traceback- the V parameter of the decoder.
- Constraint Length- the L parameter of the decoder.
- Type- '1' for serial decoder (VSAA, VSBERAA), '2' for serial/ parallel decoder (VSPAA, VSPBERAA, VSPDLAB)
- -b (optional)- this switch is used to analyze decoder simulations with BERT (selects VSBERAA when type is '1', VSPBERAA when type is '2')
- -d (optional)- this switch is used to analyze the VSPDLAB decoder.

Running the utility without a complete parameter list displays the required inputs. When a valid input list is given to the program, it displays the following outputs:

```
Reading Simulation File

Writing Results File

Number of Errors Found is X

Output BER is Y
```

Where *X* and *Y* are the results calculated during analysis of the test case output.

The following files are output by **VSPTBLAB**:

- **ERRLOCSA.TXT**- contains the location of errors in the decoded output, compared to the original symbols in the **TRANSBIT.TXT** file.
- **DECBIT.TXT**-contains the decoded bits, as determined by the decoder simulation.

Appendix A- Top Level Wrappers

For each of the five decoders, there is an unencrypted top level wrapper, which can modify the parameters of the function. As the top level wrappers are in source form, they can also be used to create the Symbol Files (**.sym**) and Include Files (**.inc**).

The names of the wrappers are:

TOP_LEVEL_VSAA.TDF

TOP_ LEVEL_VSBERAA.TDF

TOP_LEVEL_VSPAA>TDF

TOP_LEVEL_VSPBERAA.TDF

TOP_LEVEL_VSPDLAB.TDF

As an example, the source code for **TOP_LEVEL_VSAA** is shown below:

```
FUNCTION vssa (sysclk, reset, enable, load, rr [(n*softbits)..1])
RETURNS (ready, decbits [bitsout..1], outvalid, normalize;
PARAMETERS
(
n= 2
softbits= 3
L= 7
ga= 91
gb= 121
gc= 0
gd= 0
ge= 0
gf= 0
gg= 0
bmgwide= 12
v= 35
);
constant bitsout= ceil (v DIV (2^(L-1)));
subdesign top_level_vsaa
(
sysclk, reset, enable : INPUT;
load : INPUT;
rr [n*softbits..1] : INPUT;
ready : OUTPUT
decbits [bitsout..1], outvalid : OUTPUT;
normalize : OUTPUT
)
BEGIN
```

```
(ready, decbits [bitsout..1], outvalid, normalize)=

vsaa (sysclk, reset, enable, load,

rr [(n*softbits)..1])

WITH (n=n, softbits=softbits, L= L, ga= ga, gb= gb, gc= gc, gd= gd, ge= ge,
gf= gf, gg= gg, bmgwide= bmgwide,   v= v);

END;
```

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com