Start by reading the file USING XBGDP

XB GAME DEVELOPER'S PACKAGE

Juwel - PEEKV and POKEV added to the compiler. You need XB 2.9 G.E.M..to use these in XB. When using autocomplete, Escape or Fctn 9 will kill autocomplete. (Earlier versions used Fctn3)

Juwel – Assembly support has been added to the compiler. Now you can compile program written for T40XB, T80XB, The Missing Link, or XXB. Or you can write your own custom assembly routines to support a compiled XB program.

Isabella7 - Made some significant adjustments to the loader programs. The MiniMemory option has been removed. The loader no longer asks where the runtime routines are located. It reads the first line of the object code and determines which loader should be used. The loader for low memory has been modified to be compatible with the 20x faster loader in XB 2.8 G.E.M.

Isabella6 - Added the ability to delete a file on a disk drive with the XB statement DELETE "DSKn.FILE". Added boot tracking so you can have the XBGDP on DSK1 to DSK9. If it is on DSK3 you would start it up with
RUN "DSK3.LOAD". There is one case where boot tracking does not work - if you are using the TI assembler that comes with the package, the return will be to the start of XB which tries to RUN DSK1.LOAD. There is no practical way to change this. You will have to type RUN "DSK3.LOAD" again or you could have a one line LOAD program on DSK1: 10 RUN "DSK3.LOAD". This does not happen if you use Asm994a.

Isabella5 - Fixed a bug that caused a crash when inputting an array element with ACCEPT AT. Fixed a bug that caused problems passing an array element to a subprogram. When using ACCEPT AT with SIZE, the XB source program will "beep" when reaching the right hand boundary. The compiled version will 'honk". This is a side effect from using the console text editor and is not practical to change. The behavior is noted in the manual. CALL ERR is not supported by the compiler and the docs have been revised to describe this.

Isabella4 - Added Asm994a assembler from the Win994a suite. This makes it much easier to set up.

Isabella3 - Changed loader for 32K (runtime in low memory) so it doesn't overwrite unused areas of low memory. These can be used for passing values or strings when chaining programs. Slight revision to compiler docs.

Isabella2 - bug fixes to the compiler. There were still several bugs that happened when passing arrays. All of these are now fixed. IF/THEN/ELSE logic is now completely compatible with XB. A rare bug arose when DISPLAY AT was used with SIZE and more than one element in the print list. This clears the screen to row 28. PRINT, DISPLAY, INPUT prompts are so intertwined that a fix would take weeks. If this is a problem you

can concatenate the elements in the print list and display them as one string.
XB32K has been added to the package. This lets you create an XB program that is up to 32K long.

"Isabella"

COMPILER: The compiler actually does now fully support user defined subprograms, With Hondarrabi it could not pass an array to a subprogram such as:
10 A(2,3)=523::CALL TESTSUB(A(,))
100 SUB TESTSUB(Z(,))
110 PRINT Z(2,3) and 523 will be printed.
The XB loader that uses the full 32K of memory (i.e. put runtime in low memory) can now save in both EA5 and XB format.You no longer have to go to MiniMemory for that (although it is quite a bit faster to do that)
The usual minor tweaks to the code and docs.

"Hondarrabi"

COMPILER: The compiler fully supports user defined subprograms, with this difference: when using subprograms, the compiler will shorten the name to the first 6 letters. You can use longer names as long as the first six letters are not duplicates of another subprogram. CALL UPDATEWHITE and CALL UPDATEBLACK would not compile properly, but CALL UPDATWHITE and CALL UPDATBLACK would be fine.
A bug was fixed that interfered with returning to the main menu from XB256 to the menu in real iron.
CALL KEY and CALL JOYST are modified for a bit more speed. Both KEY and JOYST use the same keyscan routine built into the console. If JOYST is immediately followed by KEY then both share one keyscan which is a bit faster. If KEY is first or if an instruction is between them then each does its own keyscan.the same as XB.

"Grechetto"

COMPILER: The compiler has been extensively rewritten. It is still an XB/assembly hybrid program, but the main loops are now all in assembly which gives about a 5x speed increase.
The compiler can now handle nested arrays.
You can now assign multiple variables in a LET statement. i.e. 10 X,Y,Z=7 or 10 IF Z=7 THEN X,Y,Z=3
Speech can now be used. Both CALL SAY and CALL SPGET are supported.
A minor bug when printing an array followed by a semicolon has been fixed.
If the compiler finds a decimal point it prints NON INTEGER next to the line number and scrolls one line. This way you can go back to the XB program and easily fix the error.
There are some improvements in memory allocation.

A folder has been added with files modified to work with a real TI99. Copy GRECHETT99 to a floppy diskette and put it in drive #1.

"Frappato"

XB256 and XB have been altered so you can tell whether autosave is active. When it it active there is a light pixel in the center of the cursor. The cursor becomes solid black when autosave is no longer active.
XB256 has been altered so as to try to modify the XB grom starting at >6000. If you have enabled gram at >6000, this can addresses some stability issues that arise when using cpu overdrive.
XB256HM is gone. Instead, XB256 is modified so you can merge an XB program with the XB256 loader, making a nice neat package.
Plus the usual improvements to the docs.

"Encruzado"

COMPILER: The big change is that you now have an option to put the runtime routines into low memory from >2000 to >3FFF. This would be necessary is if your compiled program is too large to run in the 24K of high memory, with both compiled code and the runtime routines all located there. With the runtime routines in low memory it is safe to say that you should have no memory problems when compiling even the largest XB programs. If the program is saved as EA5 the program will be loaded exactly as before. If you save as an XB loader there are two programs. The first loades the low memory portion of the code and then automatically loads and runs the second program. Other than a brief pause while the second program is loaded, you will see no difference in how this functions.
When running from XB there are now three ways to start the compiled program.
CALL LINK("RUN") starts the program with a scan that breaks the program execution when F4 is pressed.
CALL LINK("RUNEA") starts the program exactly as it would be when running as EA5. No F4 scan is performed.
CALL LINK("RUNV") starts the program but without resetting any of the Screen2 patterns or character definitions. This lets you chain compiled programs together while retaining all the graphics created by the first program.
A bug in playing sound lists from EA5 was corrected.

XB256: The big change here is that there is a second version of XB256 called XB256HM.This provides a way to save an XB program and XB256 in the same program; If you don't need the speed of a compiled program, this makes a nice, neat one program package.

XB GAME DEVELOPER'S PACKAGE "Dolcetto"

COMPILER: A couple of bugs were fixed. If you broke a compiled program running from XB and if CALL LINK("FREEZE") had been used then sprite motion was turned off when you ran the program again. Under certain conditions the sound list processor

could leave one or more sound generators turned on
XB256: Both bugs described above could cause the same trouble with XB256 and have been fixed. COMPRESS had a few changes so it provides better prompts when saving data. You can turn off the autocomplete routine in XB and XB256 by pressing F3 when prompted OLD DSK.
All the documentation has been revised with extensive changes to the sound table section of the XB256 manual. SLCONVERT in particular was very poorly described and almost unuseable. Using it should be much easier now.

XB GAME DEVELOPER'S PACKAGE "Chardonnay"

COMPILER: A bug in the implementation of IF/THEN/ELSE was corrected. It worked fine except when compiling the earlier TI BASIC format. 100 IF X=7 THEN 200::Y=3::Z=4 now compiles properly.
XB256: A new subprogram has been added CALL LINK("RUNL1",A$) lets an XB program load another and run starting at line 1, *but without performing the prescan and resetting all the variables.*
An earlier version of COMPRESS was somehow included in Bordeaux and Amontillado. Chardonnay has the proper version.

XB GAME DEVELOPER'S PACKAGE "Bordeaux"

The main improvement is that now the compiler has been expanded to properly run the more versatile IF/THEN/ELSE format used by Extended BASIC. A compiled program will process these the same as in XB with one exception:
Nested IF/THEN/ELSE will not work properly. 100 IF A=1 THEN IF B=2 THEN C=3 ELSE D=4 ELSE E=5 will not crash, but will not route the program properly.

XB GAME DEVELOPER'S PACKAGE "Amontillado"

Huge upgrade to the user interface. Now everything runs from a Menu program ("LOAD"). Filenames are passed between the programs and auto complete is active for all steps in the compilation process, which means that you pretty much just need to press Enter at the prompts.
The Funnelweb assembler has been modified to be a stand alone program.
Compiler loader was modified to use the Editor Assembler utilities which include an assembly language loader program that loads in seconds, rather than the minutes required by the XB GPL language loader.
Some big rewrites in the manuals, especially the compiler documentation. It now has much more detail on how to set up to use the Asn994a cross compiler. Asm994a and the

new compiler loader remove 2 large bottlenecks in the compilation process.
From now on all upgrades will be named after a wine variety.

Start by reading XBGDPDOCS

COMPILER

November 2017 - Chased down and fixed a bug in ACCEPT AT when using SIZE(negative number) that blocked anything from being entered.

Changes in Extended BASIC compiler v2.56
The compiler has been significantly expanded to include all the XB256 assembly language extensions. XB256 is included as part of this package. It removes most of the restrictions imposed by Extended BASIC. You can toggle between two independent screens. Screen1 is the graphics mode normally used by Extended BASIC; Screen2 lets you define 256 characters, more than double the number normally available to XB. When in screen2, you can use up to 28 double sized sprites using the patterns available to Screen1. You can scroll screen characters left, right, up, or down and specify a window area for scrolling, leaving the rest of the screen unchanged. Other routines let you scroll smoothly one pixel at a time to the left, right, up or down. There are miscellaneous subroutines that let you hilight text, set the sprite early clock, print on the screen using all 32 columns, read from or write to the VDP RAM, write compressed strings to VDP, and move sound tables into VDP. Be sure to use integer arithmetic when developing programs that you intend to compile.

Changes in Extended BASIC compiler v2.12
The bug fixes above caused a serious side effect: Although strings displayed properly, numbers would not print when using DISPLAY AT with SIZE. That bug has been fixed. The compiler now gives you the option to load compiler extras. Select N if you are not using them.

Changes to Extended BASIC compiler v2.11
Several bugs have been repaired. Two more line numbers (133 and 253) were found that compiled incorrectly. HCHAR, VCHAR, and SEG$ did not work properly with a repeat value of 0. The screen timeout counter would blank the screen when playing a game even though the joysticks were being used. The keyboard did not scan properly after CALL KEY(1,K,S) or (2,K,S). DISPLAY AT with a SIZE blanked the entire line. These have all been fixed.

Changes to Extended BASIC compiler v2.1
The most important changes are the repair of several bugs. The line number bug caused certain line numbers (235,238,239,240,254) to compile incorrectly. The sprite bug crashed the computer when you deleted a sprite that was not in motion. A bug in the print routine kept lines from scrolling properly. These have all been fixed in version 2.1
The following was added to the compiler:
CALL PEEK

CALL LOAD
MAX
MIN
RPT$
The manual has been updated and expanded.

XB256

January 2018 - Added subroutine CALL LINK("RUN",A$) which is like RUN "DSK1.PROGRAM" except you can use a string variable. Fixed bug caused by another upgrade that kept the disk catalog utility from working
November 2017 - Repaired the missing CALL LINK("DELAY") so a patch to loader is no longer necessary.

XB256 Update - November 12, 2014

Added SLCONVERT which is based on SLCOMPILER. It gives more versatility when converting an XB program into a sound table. Added a disk catalog program ("CAT"). Added a fast random number generator for integers ("IRND"). Updated manual with even more information about sound tables.

XB256C update - May 18, 2014

Video memory has been remapped to put the 256 character screen in a different location. This makes XB256 compatible with the CF7, Myarc controller, etc. There is somewhat more stack space, and allocating memory for sound tables is simplified. Manual was updated with much more information about sound tables.

XB256 update - April 20, 2014

Two subprograms added: SCPXUP and SCPXDN for smooth scrolling vertically. Subprogram for "text crawl" ala Star Wars renamed CRAWL.