Version 0.2

Part I : Configuring the Saturn compiler and the SGL.

First, after you get both the Saturn GCC compiler (Cygnus based, for MS-DOS) and the leaked SGL library, you need to install them.

Unlike other development tools, GCC does not come bundled with it's own IDE (acronym for Integrated Development Editor). This means that you have to provide your own text editor. If you code in C (Saturn is not the tool to learn how to use C) you can keep using your favorite editor.
However, if you have doubts, or want to check other choices, I recommend 2 (free) alternatives:

What is GCC?

GCC is a C compiller, mainly used in *nix (Linux, Unix, FreeBSD, HP-UX) enviorments, and it's a nice and very supported tool. It's also a GNU product, meaning that it's free to use and the source is open to any developer.

RHIDE is a powerfull text-based C editor, that ressembles the Borland interface. If you want to do editing in DOS, I highly recomend this one.
EditPad products are my favorite editors for Windows. EditPad Classic has no syntax highlighting but it's postcardware and mutch faster than EditPad Lite. Combined with the Programmers IDE, it becomes a friendly dev enviroment for Windows.

You could also consider a commercial option such as UltraEdit. Not so expensive (around 20 bucks) and has sintax highlight and can compile the project and even create the ISO from the GUI if you configure it too.
(thanks to Artemio Urbina for the suggestion)

As I told before, you can edit those C files with whatever you want. I often use Visual Dev Studio (the Microsoft editor that comes bundled with their dev products) to edit, for example. Just DON'T USE COMPILE/MAKE EXE. That is an issue that needs to be taken in consideration separately.
GCC relies on a utility called MAKE to err… make an executable. Altough it's out of the scope of this text, you should read about make and makefiles. For now it suffices to say the SGL spares a good amount of distress, by allowing us to just edit a tiny text file called OBJECTS and use a pre-made MAKEFILE.
Now, off to the good part. Installing.
Decompress the compiller to a directory, let's say C:\dev\. It will instantly create a SATURN directory. Good. That wasn't very hard, was it?
Now, decompress the SGL files, in a directory like this C:\dev\SATURN\Sgl\. Unlike the compiller, you need to add a Sgl directory.

(The follwing procedure is mentioned twice on the README directory. Obviously, you missed it. Go, read)

Now, off to the magic file called SETENV.BAT. You should have one already done, and looks something like this:

WARNING ABOUT SGL

The SGL is a leak of the official Sega Development Library. That said, you need to be an official Saturn game developer to be able to use it. As SEGA does not accept Saturn game developers anymore, and has offically stopped to support the Saturn console, it's a thin ice area. This means that Sega might try and force us to drop the support, but has no real reason do to so. Anyway, everything you do with the SGL is potencially illegal, okay?

```
REM sh-hms
SET PATH=C:\SATURN\BIN;c:\windows\command
SET GCC_EXEC_PREFIX=C:\SATURN\LIB\
SET INFOPATH=C:\SATURN\INFO
SET C_INCLUDE_PATH=C:\SATURN\include
SET CPLUS_INCLUDE_PATH=C:\SATURN\include\cxx;C:\SATURN\include
SET GO32=EMU C:\SATURN\BIN\EMU387
REM Set TMPDIR to point to a ramdisk if you have one
SET TMPDIR=C:\TMP
```

Change the files so that they point to the correct places, in our case, it would be:

```
REM sh-hms
SET PATH=C:\dev\SATURN\BIN;c:\windows\command
SET GCC_EXEC_PREFIX=C:\dev\SATURN\LIB\
SET INFOPATH=C:\dev\SATURN\INFO
SET C_INCLUDE_PATH=C:\dev\SATURN\include
SET CPLUS_INCLUDE_PATH=C:\dev\SATURN\include\cxx;C:\Gamez\SATURN\include
REM Set TMPDIR to point to a ramdisk if you have one
SET TMPDIR=C:\TMP
```

(Notice I erased the SET GO32 file. If you want to compile on a old 486 or a 386, keep it)

Now, add some SGL paths:

```
REM sh-hms
SET PATH=C:\dev\SATURN\BIN;c:\windows\command
SET GCC_EXEC_PREFIX=C:\dev\SATURN\LIB\;C:\dev\Saturn\
SET INFOPATH=C:\dev\SATURN\INFO
SET C_INCLUDE_PATH=C:\dev\SATURN\include;C:\dev\SATURN\Sgl\include;
C:\dev\SATURN\Sgl\Inc
SET CPLUS_INCLUDE_PATH=C:\dev\SATURN\include\cxx;C:\Gamez\SATURN\include;
C:\dev\SATURN\Sgl\include;C:\dev\SATURN\Sgl\inc
REM Set TMPDIR to point to a ramdisk if you have one
SET TMPDIR=C:\TMP
```

(notice that some lines wrap around in this text)

And add the following SET LIBRARY line for the Sgl library files (not needed, but just in case)

```
REM sh-hms
SET PATH=C:\dev\SATURN\BIN;c:\windows\command
SET GCC_EXEC_PREFIX=C:\dev\SATURN\LIB\;C:\dev\Saturn\
SET INFOPATH=C:\dev\SATURN\INFO
SET LIBRARY_PATH=C:\dev\SATURN\lib;C:\dev\SATURN\Sgl\lib
SET C_INCLUDE_PATH=C:\dev\SATURN\include;C:\dev\SATURN\Sgl\include;
C:\dev\SATURN\Sgl\Inc
SET CPLUS_INCLUDE_PATH=C:\dev\SATURN\include\cxx;C:\Gamez\SATURN\include;
C:\dev\SATURN\Sgl\include;C:\dev\SATURN\Sgl\inc
REM Set TMPDIR to point to a ramdisk if you have one
SET TMPDIR=C:\TMP
```

Now, open a MS-DOS command line, head for the directory and type SETENV.BAT. This sets up the enviroment for the GCC compiller.

The final test. Enter GCC in the command line. It should report "gcc.exe: No input files". If not, review all the steps and check the box next to this text.

"Out of environment space" error

No problem. On the SETENV.BAT directory, create a shortcut to C:\Windows\Command.com. Then chose Proprietes, select the Memory tab and set the Initial Enviroment from Auto to 1024. If you go to the Programs tab and add SETENV.BAT in the Batch File, by double clicking in the shortcut you created, you enter the GCC enviroment automagicly.

Let's compile somethiing, then:
Head to the Sgl/Samples directory via that command line and pick one directory, for example Akira. Type MAKE, and it will provably give you "make.exe: Nothing to be done for `all'". Type MAKE CLEAN, and you'll erase any intermediate files and outputs (it's safe to do so, no source code is lost). Write MAKE again and you'll see the compile process.

```
        gcc ../common/cinit.c  -O2 -m2 -g -c -I../../inc -o ../common/cinit.o
        gcc main.c  -O2 -m2 -g -c -I../../inc -o main.o
        gcc aki_ashi.c  -O2 -m2 -g -c -I../../inc -o aki_ashi.o
        gcc aki_dou.c  -O2 -m2 -g -c -I../../inc -o aki_dou.o
        gcc aki_kao.c  -O2 -m2 -g -c -I../../inc -o aki_kao.o
        gcc aki_kata.c  -O2 -m2 -g -c -I../../inc -o aki_kata.o
        gcc aki_ude.c  -O2 -m2 -g -c -I../../inc -o aki_ude.o
        gcc aki_walk.c  -O2 -m2 -g -c -I../../inc -o aki_walk.o
        gcc workarea.c  -O2 -m2 -g -c -I../../inc -o workarea.o
        gcc -m2 -L../../lib -Xlinker -T../common/sl.lnk -Xlinker -Map -Xlinker sl.map –
X linker -e -Xlinker ___Start -nostartfiles  ../common/cinit.o main.o aki_ashi.o a
ki_dou.o aki_kao.o aki_kata.o aki_ude.o aki_walk.o workarea.o -lsgl -o sl.coff
objcopy -O binary sl.coff sl.bin
```

Voilá. You compiled your first C SGL sample. I hope. Check all the steps if not.

The next lesson will speak about your first Saturn C SGL program.