Jeffrey Quinn

Software Sega CD Simluator

The project is a piece of software named "AGES." It is the development of a simulator of a Sega CD, a peripheral add-on to a Sega Genesis that includes a CD-drive, a M68000, a custom VDP (video display processor) chip, a PCM (pulse code modulation) sound chip, and additional RAM. It will be added to a set of already existing code that simulates the Sega Genesis and another add-on, the Sega 32X.

Inputs and Outputs of the project:

The general inputs and outputs for AGES are listed below. Refer to Fig. 1 for a graphical representation.

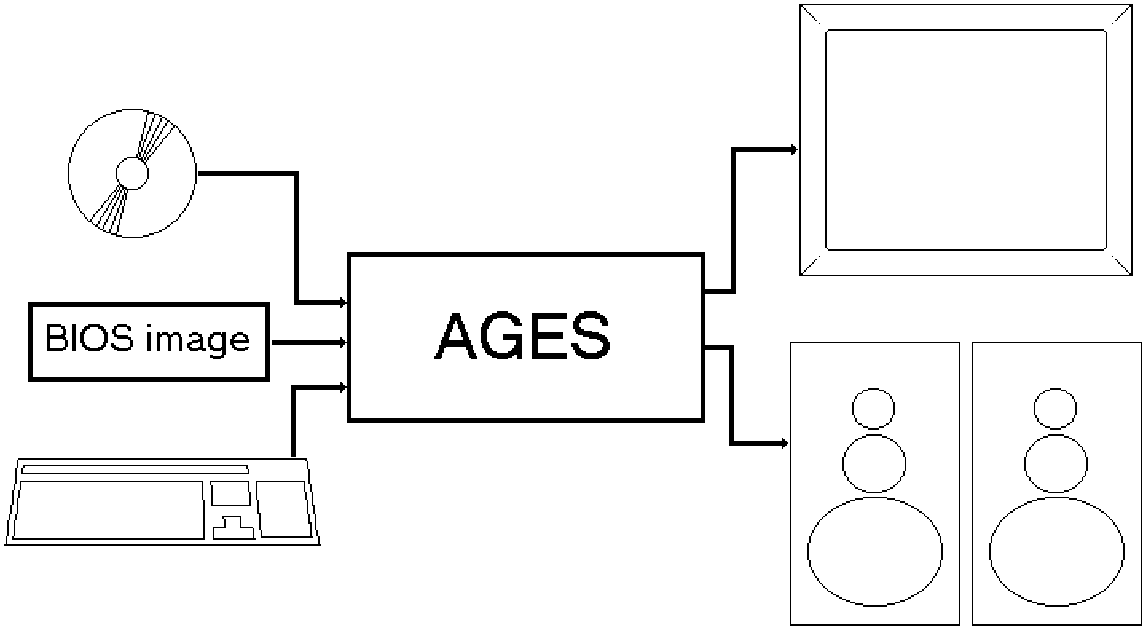
Inputs:

- Game control inputs
- Actual game CD
- One binary image of the Sega CD BIOS software

Outputs:

- Game visuals
- Game sounds/music

Fig. 1: AGES inputs and outputs.



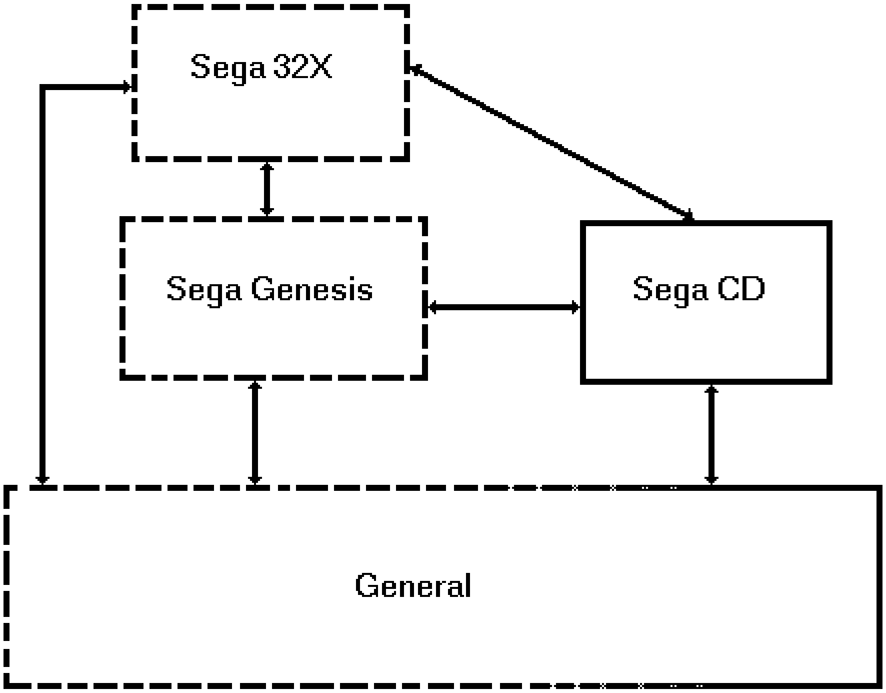
Description:

The user will be able to place a Sega CD game disc into the CD-drive of his/her computer, after retrieving a binary image of the Sega CD BIOS software, and interact with the game software in the same way that he/she would by using a real Sega CD machine. This means that the picture will be displayed in its entirety on the computer monitor (instead of television), the sound will be reproduced in its entirety via the computer's sound card and speakers (instead of television speakers), and the input will be received via the keyboard or an attached gaming device (joystick) at the user's discretion instead of the gaming device (joystick) native to the Sega CD machine. Aside from these slight differences, the behavior of the software shall attempt to duplicate the behavior exactly. One notable exception is the speed of the drive. A real Sega CD's drive was "single speed," however, the simulation software will utilize the speed of the PC CD-drive to provide faster performance (E.G. shorter "load times.")

Software Modules of the project:

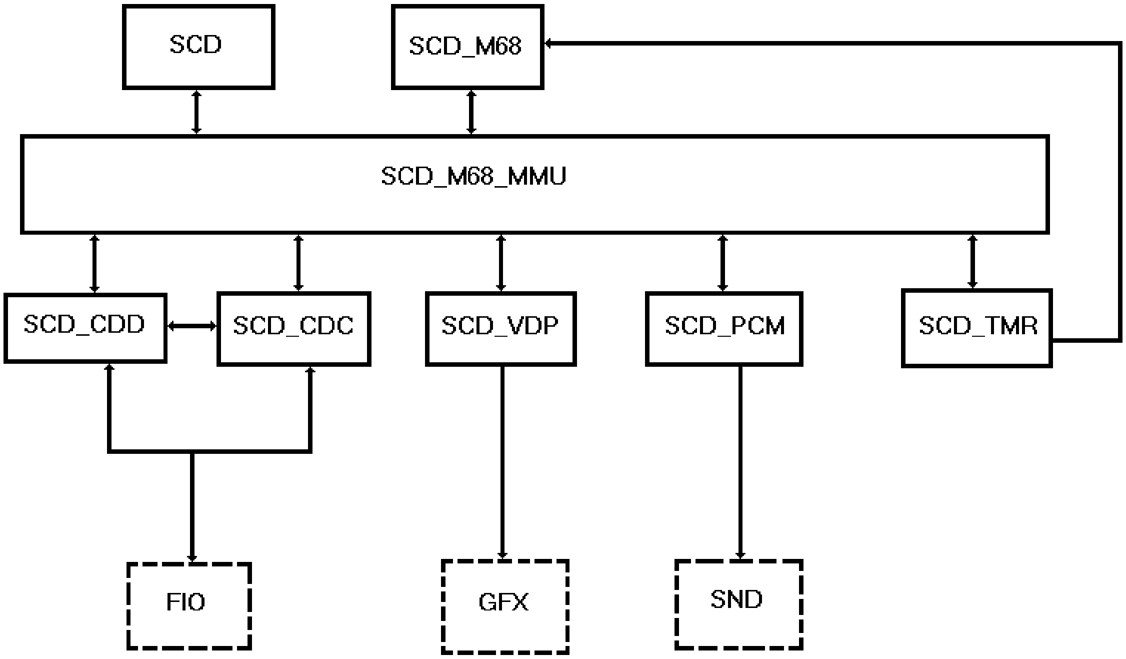
AGES is divided into four different blocks, the Sega Genesis Block, the Sega 32X Block, the Sega CD Block, and the General Block. The first two blocks are written already and are beyond the scope of the project. The General Block also has been written, but is not beyond the project's scope as it will require additions to be made to it to accommodate new functionality needed by the Sega CD Block. The primary focus of the project is the Sega CD Block itself. Refer to Fig. 2 for an illustration of the blocks. The dashes indicate what is already completed and beyond the scope of the project.

Fig. 2: AGES Block Diagram.



AGES is further broken down into two types of modules, “chip modules” and “engine modules.” Each chip module is a subset of the simulation containing all code pertaining to a single chip and a small amount code that handles the interaction of said chip with other chips. Engine modules handle interaction with a specific part of the PC hardware. The first three blocks contain only chip modules. The General Block contains only engine modules. AGES already has fully written code for all chips located in the Sega Genesis and Sega 32X, and as previously stated, these are beyond the scope of the project. AGES also has a complete set of fully written engine modules. This project is concerned only with the chip modules of the Sega CD and adding code to the engine modules as needed. No new engine modules will be created. Refer to Fig. 3 for an illustration of the all modules pertaining to the project and how they relate to each other.

Fig. 3: AGES modules in project scope.



Sega CD chip modules:

M68_EMU This module simulates the Motorola 68000 CPU chip. It is not strictly a Sega CD module, but rather is a general module that is instantiated twice, once as GEN_M68, and once as SCD_M68. It has three inputs and three outputs for accessing memory. The inputs are read data in byte, word, and dword sizes. The outputs are write data in byte, word, and dword sizes. Additionally, it has one input for IRQ level. This module has already been written and debugged.

SCD

This simulates the system registers of the Sega CD, and contains basic driver information specific to the Sega CD. The inputs and outputs of this module are the data stored in the registers. This module is not yet written.

SCD_M68_MMU

This simulates the memory map of the M68000 located in the Sega CD. This modules takes inputs and outputs from each other module in the Sega CD block. This module is not yet written.

SCD_CDC

This simulates the behavior CD controller in the Sega CD. The controller is used for reading data from CD data tracks. This module inputs and outputs the data in the sixteen 8-bit CDC registers. Additionally, this module takes an input from the SCD_CDD. The input is a periodic “read” signal asserted when the CDD is playing a data track to be decoded and error corrected. This module is not yet written.

SCD_CDD

This simulates the behavior of the physical CD drive in the Sega CD. The drive is used for activating the spindle motor, reading TOC info, playing tracks, etc… Its inputs are data in the ten 4-bit Transmit Command Registers. Its outputs are data in the ten 4-bit Receive Status Registers and a periodic “read” signal (sent to the SCD_CDC when a data track is being played). This module is not yet written.

SCD_PCM

This simulates the behavior of the PCM (sound) chip located in the Sega CD. Its inputs and outputs are the data in the PCM registers. This module is not yet written.

SCD_VDP

This simulates the behavior of the VDP (graphics) chip located in the Sega CD. Its inputs and outputs are the data in the VDP registers. This module is not yet written.

SCD_TMR

This simulates all the timer chips located in the Sega CD. Its inputs and outputs are the data in the timer registers and an IRQ level. This module is not yet written.

Engine modules:

GFX

This handles all interfacing with the host PC’s graphics hardware. This module inputs raw[1] graphics data from all VDP simulation modules and converts it to a form useable by DirectX for outputting to the monitor. This module is already written, but will require additional code to support the needs of the Sega CD simulation modules.

[1] In whatever format is native to the chip delivering the data.

SND This handles all interfacing with the host PC’s sound hardware. This module inputs raw[2] sound data from all sound simulation modules and converts it to a form useable by DirectX for outputting to the soundcard and speakers. This module is already written, but will require additional code to support the needs of the Sega CD simulation modules.

FIO This handles all interfacing with the host PC’s file/IO hardware. (E.G. image files off the harddrive or direct use of the CD drive) It inputs raw[3] file/IO data and converts it to a program-wide consistent binary format for outputting to the various modules. This module is already written, but will require additional code to support the needs of the Sega CD simulation modules.

[2] In whatever format is native to the chip delivering the data.
[3] In whatever format is native to the chip delivering the data.