Changes

Date	Version	Description
9/8/2014	1.0	Clarified the shuffle algorithm.
9/22/2014	1.1	Modified Random Number Generator section to reflect Windows OS cryptographic RNG.
10/13/2014	1.2	Updated directory structures and partition names.
11/3/2014	1.3	Updated the Non-volatile and Critical Memory section.
12/3/2015	1.4	Cleanup for submission of Ba Baccarat.
1/4/2016	1.5	Added feature package section.
6/13/2016	1.6	Cleaned up for BMM.
4/18/2017	1.7	Add 16G SATADOM structure.

CONFIDENTIAL
Casino Game Maker, Inc.
369 North 100 West, Suite #8
Cedar City, UT 84721

Valhalla Platform Technical Manual
Version 1.7

Overview

The Valhalla platform is a Windows Embedded class II/III gaming platform based on C++ for the core and AS3/Starling for the game engine. Both the core and game engine are designed in a way to allow for quick game changes using a data driven approach.

Core

General:

- Robust C++ Engine (Designed using cross platform components for the future.)
- Hardware Independent Software. Uses off the shelf motherboards.
- Locked down SATADOM for all regulated software components.
- Large NVRAM area (mSATA mini) with required validity and redundant checks.
- Tablet User Interface Support
- XML/Data Driven
  - Customized Service Screens
  - Customized IO Configurations for Various Cabinets

Server Protocols & Ports:

- SAS Port for Slot Accounting System
- Feature Server (ETG, Tournament, & Progressives)

Peripherals:

- MEI Bill Acceptor
- JCM Bill Acceptor
- FutureLogic Printer
- Ithaca Printer
- USB IO Board

Game Engine

General

- Multiple Targets
  - Cabinet / Desktop (Development)
  - Standalone (Development and Demo) PC, Mac, Android, iPad / iPhone Devices
- GPU Supported Engine (Starling/Away3D) 60 FPS Standard
- Baccarat, Roulette, and Slot Engines
- Physics Engine and 3D Real Play
- Multiple Bonus Types – Reveal, Matching, etc.
- Gamble Features
- Various Free Spin Types
- Multiple Resolutions and Layouts, including HD and 4K
- Top Screens with Progressives
- XML/Data Driven
- Flexible Particle Engine
- Celebration Animations

CONFIDENTIAL
Casino Game Maker, Inc.
369 North 100 West, Suite #8
Cedar City, UT 84721

Media Structure & Layout

Media Devices

Device	Description	Manufacturer
8G or 16G SATADOM	OS, Core, Game Engine, & Game	Innodisk
8G mSATA Mini-SATA	NVRAM	Innodisk

SATADOM File & Directory Structure

#	Item	Description
1	Windows Embedded 7 files and directories.	OS
2	C:\CGM\Boot*	Boot loader files.
3	C:\CGM\Core*	Core program and configuration files.
4	C:\CGM\Game*	Game engine and assets. Topper engine and assets, if applicable. User interface engine and assets.
7	C:\CGM\Utilities*	Utility programs (e.g., Touch screen calibration).

8G SATADOM Physical Structure

#	Item	Size (Bytes)	Address
1	OS, Core, Game Engine and Game Partition (C:\)	8,009,023,488	1,048,576
2	Verification Hash (Varies if encrypted)	20	8,010,072,064
3	Unused Area	Various	8,010,072,084

16G SATADOM Physical Structure

#	Item	Size (Bytes)	Address
1	OS, Core, Game Engine and Game Partition (C:\)	16,011,755,520	1,048,576
2	Verification Hash (Varies if encrypted)	20	16,012,804,096
3	Unused Area	Various	16,012,804,116

Mini-SATA File & Directory Structure

#	Item	Description
1	D:\Var	Logs, Meters, Critical Memory, etc.
2	D:\Feature_Packages	Game Specific Feature Packages

CONFIDENTIAL
Casino Game Maker, Inc.
369 North 100 West, Suite #8
Cedar City, UT 84721

Boot Process

1. When the OS boots, the first program to run is Cabinet.exe. Using the SHA-1 algorithm with an HMAC seed, this program will hash the entire 1st partition of the SATADOM. The HMAC seed is defined as “Innovation In Gaming”.
2. Once the hash is generated, it is compared with the 20 byte hash stored in the area following the SATADOM.
3. If there is a mismatch, an error is shown and the boot process is halted.
4. If there is no error, the boot process continues and runs the core and game programs.

Non-volatile and Critical Memory

All non-volatile memory is stored on a mSATA device in the *D:\Var* directory as two files, a primary and secondary, for each memory area. For example, the meters are stored as *Meters_P.dat* and *Meters_S.dat*. Each file contains a 32 bit CRC in the last four bytes of the file. This provides a method to verify the integrity of the contents.

Also, each file has access permissions that only allow the core to modify or delete it. In other words, somebody accessing the OS from the outside would not be able to modify or delete these files.

All non-volatile memory is checked on startup. If the primary file is valid, it is used to restore the associated object. If a failure occurs in the primary file and the secondary file is valid, then the secondary file is used to restore the associated object. If both the primary and secondary files fail, then an error message is displayed and the boot process stops.

Additionally, all critical memory is checked when all doors have been closed.

The method to read/upload the critical memory to/from an EGM is as follows:

- Power down the EGM.
- Remove the mSATA device from the EGM.
- Using a mSATA to USB reader, copy the files to/from the mSATA device.
- Replace the mSATA device.
- Apply power to the EGM.

RAM Clear Process

The RAM Clear USB device contains a special setup directory structure. When the EGM is booted, it detects this structure and removes all or part of the files in the *D:\Var* directory as necessary. The process is as follows:

1. Power off EGM.
2. Open slot and logic doors.
3. Insert RAM Clear USB device into 9 pin connector on side of logic box.
4. Power on EGM and wait for the RAM clear process to finish.
5. Power off EGM and remove the RAM Clear USB device.
6. Close slot and logic doors.
7. Power on EGM.

CONFIDENTIAL
Casino Game Maker, Inc.
369 North 100 West, Suite #8
Cedar City, UT 84721

Feature Packages

In order to facilitate a variety of assets for different approved engines, a feature package process has been implemented in the Valhalla platform. A feature package is roughly defined as an asset or assets that will modify the look and/or feel of a game and is not required to have a regulatory signature. This does not remove the requirement of the assets being cursory checked by a regulatory body.

The process is as follows:

1. Power off EGM.
2. Open slot and logic doors.
3. Insert Feature Package USB device into 9 pin connector on side of logic box.
4. Power on EGM and wait for the process to finish.
5. Power off EGM and remove the Feature Package USB device.
6. Close slot and logic doors.
7. Power on EGM.

The Feature Package assets are contained in the USB device as a single compressed file. When the installation occurs, this file is uncompressed and copied to the E:\ drive as follows:

E:\Feature_Packages*game_name**assets*

For example, a feature package Keno_Holiday_1.7z → E:\Feature_Packages\Keno\...

In addition to the actual assets, a simple signature file associated to the feature package is copied over. This signature file includes an HMAC-SHA1 hash that will be used to validate the feature package files when the EGM is booted. If a failure occurs, the EGM boot process will halt and a message will be displayed. Either the feature package will have to be removed or re-installed to fix this failure.

Settings.xml File

The Settings.xml file contains all the available settings for the platform. Each setting has the following attributes available:

Name	The symbolic name of the setting. This is used internal in the core.
Description	The English name if the setting. This is used in the service menu.
Type	The type of the setting. (STRING, INTEGER, LIST, etc.)
Minimum	The minimum value that it can be set to.
Maximum	The maximum value that it can be set to.
Default	The default value that it will be set to after a RAM clear.
List_Names	The English names of all items in list. (CSV)
List_Values	The values of all items in list. (CSV)
Lock	If true, locks the setting after a RAM clear.
Enabled	If true, displays the setting in the service menu.

The following settings are available:

System Settings	
ASSET_NUMBER	Asset number of EGM for the slot accounting system.
MACHINE_NUMBER	Machine number of EGM for the slot accounting system.
SERIAL_NUMBER	Serial number of EGM for the slot accounting system.
FLOOR_LOCATION	Floor location of EGM for the slot accounting system.
SAS_ADDRESS	Polling address of EGM for the slot accounting system.
VALIDATION_TYPE	Ticket validation type of EGM for the slot accounting system.
FUNDS_TRANSFER_TYPE	Funds transfer type of EGM for the slot accounting system.
BONUS_TYPE	Bonus type of EGM for the slot accounting system.
TICKET_REDEMPTION	Ticket redemption type of EGM for the slot accounting system.
RESTRICTED_TICKETS_OUT	Enables/disables restricted (AFT) tickets from being printed.
PARTIAL_TRANSFERS	Enables/disables partial AFT fund transfers.
HOST_CASHOUT	N/A
PROG_ADDRESS	N/A
Limit Settings	
CREDIT_LIMIT	Maximum allowed credit on the EGM.
JACKPOT_LIMIT	Maximum allowed win on the EGM.
FUNDS_TRANSFER_LIMIT	Maximum allowed amount that can be transferred to/from EGM via AFT/EFT.
BILL_IN_LIMIT	Maximum allowed bill that can be accepted by bill validator.
TICKET_IN_LIMIT	Maximum allowed ticket that can be accepted by bill validator.
TICKET_OUT_LIMIT	Maximum allowed ticket that can be printed by ticket printer.
MONEY_IN_LIMIT	Maximum allowed money in.
SYSTEM_TIMEOUT	Number of seconds to wait until a communication failure occurs. 0 = Forever
Peripheral Settings	
BILL_VALIDATOR	Enables/disables the bill validator.
TICKET_PRINTER	Enables/disables the ticket printer.

Game Settings

VOLUME	Overall master volume.
ATTRACTION_TIMEOUT	Number of seconds to wait in idle until the game goes into an attraction mode.
GAMBLE_FEATURE	Enables/disables the gamble feature if available.
MINIMUM_BET	The minimum allowed bet/wager, if implemented.
MAXIMUM_BET	The maximum allowed bet/wager, if implemented.
BET_TIME	The bet/wager time of the game, if implemented.
CHIP_1_DENOM	The denomination of the first chip, if implemented.
CHIP_2_DENOM	The denomination of the second chip, if implemented.
CHIP_3_DENOM	The denomination of the third chip, if implemented.
CHIP_4_DENOM	The denomination of the fourth chip, if implemented.
CHIP_5_DENOM	The denomination of the fifth chip, if implemented.
CHIP_6_DENOM	The denomination of the sixth chip, if implemented.

Config.xml File

The Config.xml file contains all the available configurations for the platform. The following configurations are available:

Core/Game	
NVRAM_LOCATION	The location where the non-volatile memory resides.
GAME_EXECUTABLE	The location and filename of the game executable.
USER_INTERFACE_EXECUTABLE	The location and filename of the user interface executable.
TOP_EXECUTABLE	The location and filename of the top screen executable.
BINGO	Enables/disables the bingo element if the game, if implemented.
RNG_DEBUG	Enables/disables RNG debug output.
IO_DEBUG	Enables/disables IO board debug output.
DEMO_CREDIT_AMOUNT	The amount of credit to give in various demo modes.
SAS_EXTENDED_GAME_ID	The extended game ID of EGM for the slot accounting system.
TICKET_DEMO	Enables/disables the ticket demo. During this mode, the bill validator will accept special tickets that the ticket printer prints. The ticket amount is embedded in the validation number.
DEVELOPMENT	Adds credit when the cash out button is pressed and no credit exists.
CURRENCY	The currency of the EGM. (US$ or HK$)
Peripherals	
BILL_VALIDATOR	Enables/disables the bill validator.
PRINTER	Enables/disables the ticket printer.
IO_BOARD	Enables/disables the IO board.
SYSTEM_INTERFACE	Enables/disables communication to the slot accounting system.
USER_INTERFACE	Enables/disables communication to the user interface.
TOP	Enables/disables communication to the top screen.
Serial Communication	
BILL_VALIDATOR_PORT	Communication port for the bill validator.
PRINTER_PORT"	Communication port for the ticket printer.
IO_BOARD_PORT"	Communication port for the IO board.
SYSTEM_PORT"	Communication port for the slot accounting system.
GAT_PORT"	Communication port for the GAT interface/tool.
Server	
SERVER_ADDRESS	Socket address of the feature server.
GAME_PORT	Socket port of the game.
TOP_PORT	Socket port of the top screen.
USER_INTERFACE_PORT	Socket port of the user interface.
DEBUG_INTERFACE_PORT	Socket port of the debug interface/injection tool.
TOP_INTERFACE_PORT	Socket port of the top interface/marketing port.
BINGO_INTERFACE_PORT	Socket port of the bingo server.

Logs

EVENT_LOG_SIZE	Number of entries in the event log.
DOOR_LOG_SIZE	Number of entries in the door log.
BILL_LOG_SIZE	Number of entries in the bill log.
VOUCHER_IN_LOG_SIZE	Number of entries in the voucher in log.
VOUCHER_OUT_LOG_SIZE	Number of entries in the voucher out log.
TRANSACTION_LOG_SIZE	Number of entries in the transaction log.
TILT_LOG_SIZE	Number of entries in the tilt log.
BONUS_LOG_SIZE	Number of entries in the bonus (AFT/Legacy) log.
PROGRESSIVE_LOG_SIZE	Number of entries in the progressive log.
GAME_LOG_SIZE	Number of entries in the game log.

Random Number Generator

The Valhalla platform uses the Windows 7 OS cryptographic random number generator. See RNG.cpp and RNG.hpp for details. The OS function CryptGenRandom is used to obtain 4 bytes of random data. These values are casted to a 32 bit number and then limited to the required scale. No external seeding is added for entropy.

Once running, the RNG will spin in the background. Every 20 milliseconds, a random number will be selected. Depending on this value, either 1 or 2 more random numbers will be selected. This effectively calls the RNG 100~150 times a second.

The game has access to the RNG via XML packets. The game is currently connected to the Core via an XML socket. Request/reply packets are sent/received with required RNG information.

For example, a 52 deck of cards would use the following algorithm to shuffle:

8. The deck is initialized from 0~51.
9. 51 random numbers are requested from the RNG. The first would be from 0~51, the next 0~50, etc.
10. Each card in the deck is swapped with the card at the index defined by the random number. An index pointer points to the offset into the deck that the swapping will begin.
11. The index pointer for the next card would be incremented.

DECK: 0 1 2 3 4 5 6 7 … 48 49 50 51

First random number: 5

DECK: 5 1 2 3 4 0 6 7 … 48 49 50 51

Second random number: 49

DECK: 5 50 2 3 4 0 6 7 … 48 49 1 51

…

Power Requirements

This table represents the general power requirements for each part of the platform.

Item	Voltage Requirements	Power Requirements
CPU using Pico PSU or Equivalent	12V @ ~16A	~192W
Tablet (TBD)	5V @ 2A	10W
Bill Acceptor (MEI SC6607)	24V @ ~3A	~72W
Printer (FutureLogic GEN2 Universal)	24V @ ~3A	~72W
Player Tracking Back-end Interface Card Reader Touch Screen	TBD	~100W
Progressive (Paltronics MIC Board)	12V @ 2A	24W
USB Charger	5V @ 1A	5W
Misc. for IO, buttons, etc.	12V @ 2A	24W
	Total	~500W

5V @ 3A = 15W
12V @ 20A = 240W
24V @ 6A = 144W