NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DESIGN PROPOSAL FOR A HIGHLY ROBUST PERIPHERAL INPUT DEVICE SWITCH FOR A MULT-LEVEL SECURE SYSTEM

by

Douglas C. Tanner

March 2012

Thesis Advisor:	George Dinolt
Second Reader:	John Mildner

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704–0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704–0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** March 2012	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Design Proposal for a Highly Robust Peripheral Input Device Switch for a Multi-Level Secure System		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Tanner, Douglas C			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943–5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number _____N/A_____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

A number of commercial vendors have tried to develop peripheral input device switches to provide high robustness, but most fail to achieve the assurance level necessary for use in a multi-level secure system. This paper provides the groundwork for designing a highly robust peripheral input device switch for Universal Serial Bus (USB) keyboards and mice by defining the requirements, the external and internal interfaces, the data flows, and the state diagrams of the switch. All of these are used to show that only a single computer connected to the switch is allowed to communicate with the attached keyboard and mouse at any given point in time, that a "Flush" command must precede any connection between an attached computer and the keyboard and mouse, and that no two CPPs are allowed to communicate with each other through the switch.

14. SUBJECT TERMS KVM Switch, USB KVM Switch, High Robustness, Multi-Level Secure, Secure Component Development, Information Assurance Design			**15. NUMBER OF PAGES** 91
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540–01–280–5500 Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

DESIGN PROPOSAL FOR A HIGHLY ROBUST PERIPHERAL INPUT DEVICE SWITCH FOR A MULTI-LEVEL SECURE SYSTEM

Douglas C. Tanner
Civilian, Department of the Navy
B.S., Presbyterian College, 2005

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2012

Author: Douglas C. Tanner

Approved by: George Dinolt
Thesis Advisor

John Mildner
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A number of commercial vendors have tried to develop peripheral input device switches to provide high robustness, but most fail to achieve the assurance level necessary for use in a multi-level secure system. This paper provides the groundwork for designing a highly robust peripheral input device switch for Universal Serial Bus (USB) keyboards and mice by defining the requirements, the external and internal interfaces, the data flows, and the state diagrams of the switch. All of these are used to show that only a single computer connected to the switch is allowed to communicate with the attached keyboard and mouse at any given point in time, that a "Flush" command must precede any connection between an attached computer and the keyboard and mouse, and that no two CPPs are allowed to communicate with each other through the switch.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLAN

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACK	Acknowledgement
IOS	Basic Input Output System
CAC	Common Access Card
COTS	Commercial Off-the-Shelf
CPP	Computer Peripheral Port
CSM	Core Switch Module
CSPLIT	Complete Split
EAL	Evaluation Assurance Level
EM	Emulation Module
DVI	Digital Video Interface
HID	Human Interface Device
IPPG	Input Peripheral Port Group
IT	Information Technology
KVM	Keyboard, Video, Mouse
MSC	Message Sequence Chart
NAK	Negative Acknowledgement
NIAP	National Information Assurance Partnership
NYET	No Response Yet
OB1	One Box One Wire
PP	Protection Profile
PSS	Peripheral Sharing Switch
SSPLIT	Start Split
TCB	Trusted Computing Base
USB	Universal Serial Bus
VGA	Video Graphics Array

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First and foremost I offer my sincerest gratitude to my thesis advisor, Dr. George Dinolt, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way.

Second, I would like to also thank my second reader, John Mildner, who pushed me to expand my views and kept me on track throughout the entire process.

I would also like to thank the SPAWAR Systems Center Atlantic for providing me the opportunity to obtain my master's degree through the Naval Postgraduate School.

Finally, I would like to thank my wife for her love, support, and encouragement without which, I would have been unable to complete my master's degree.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PURPOSE

Since the inception of classified information, individuals and organizations needed to be able to access information at multiple classification levels (domains). This has been accomplished over the years by using separate information systems for each domain, with each system having its own set of peripheral devices. By separating the domains physically, one was assured that data of a higher classification would not be spilled to domain of a lower classification. If a user requires access to several different security domains, a large workspace is needed to accommodate the keyboard, video display, and mouse associated with each information system.

In an effort to minimize the amount of equipment needed to access multiple domains from a single workspace, initiatives like the One Box - One Wire (OB1) program are currently under development [1]. The OB1 program allows multiple domains to be built into one box. This works toward the solution of minimized equipment, but it does not solve the problem of peripheral devices such as mice, keyboards, and monitors that are used as input/output devices. The current system setup still involves a mouse, keyboard, and monitor for each domain installed in an OB1 system. Figure 1 depicts an over exaggerated view of what a desktop with multiple systems connected to different domains would look like.

Figure 1. Desktop Before Implementing a KVM switch[1]

[1] Image available at http://www.blackbox.co.uk/images/technical/techoverviews/kvm-switch_before.jpg.

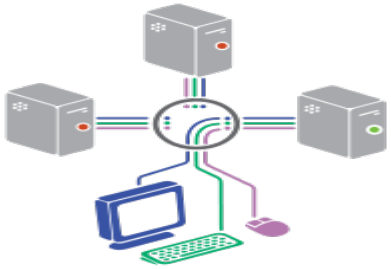

Figure 2. Basic KVM Illustration[2]

The use of a Keyboard, Video, and Mouse (KVM) Switch provides a solution to overcrowded desk space. A KVM, or Peripheral Sharing Switch (PSS) [2], allows a single set of human interface devices to be shared among two or more computers. A set of human interface devices could consist of a Universal Serial Bus (USB) keyboard, a USB mouse, and a Digital Video Interface (DVI) or Video Graphics Array (VGA) monitor. However, a KVM can also provide ports to attach input and output audio devices, other USB devices such as but not limited to printers, common access card (CAC) readers, and flash memory, and serial keyboards and mice.

Traditional KVMs are available commercially and are commonly used today in configurations that share peripheral input devices across multiple systems of the same classification level. However, there are very few solutions that are able to or attempt to attain a certification level that is acceptable in a system with multiple computers with differing classification levels due to the high cost and robust design processes that must be followed to obtain an acceptable certification level.

B. OBJECTIVES

The objective of this paper is to lay the groundwork for building a secure KVM capable of working in a multi-level secure environment. This will be accomplished by the following:

- Compare several products that are commercially available and identify possible weaknesses in these products.

[2] Image available at http://en.wikipedia.org/wiki/KVM_switch

- Derive requirements of a secure KVM to ensure no loss of confidentiality or integrity.
- Create a design proposal to include potential improvements of current technology.

C. SCOPE

A basic KVM allows a single monitor, mouse, and keyboard to be used to switch between multiple computers at the push of a button. However, the research contained within this paper will concentrate only on switching the keyboard and mouse connection between the attached computers.

Video switching capability is outside the scope of the work described here. The video portion needed to complete the KVM will be assumed to be a black box that will securely switch between multiple security domains without loss of confidentiality or integrity. The capability to share audio input/output devices will also be outside the scope of this research paper.

D. METHODOLOGY

The Twin Peaks Model "intertwines requirements and architectures to achieve incremental development and speedy delivery" [3]. This method of development provides improvements over the Waterfall Model [4] and the Spiral Life-Cycle Model [5] by allowing the system to evolve without anchoring the design to rigid requirement documents that may not consider the architecture of the system and decreasing the amount of time to move from the requirements phase to the production phase. Using the Twin Peaks approach, the requirements were refined based on architectural decisions. Some of the architectural decisions were based on security concerns, but others evolved based on the user's experience with other KVMs.

To understand the core requirements of the switch, background information is required such as what defines a KVM switch, a detailed review of the USB specification, and comparisons to commercially available KVMs. The core requirements lead to derived requirements and design choices as the internal and external interfaces and different components of the switch are defined. As each component is defined, a state

diagram is included to show the reader the functions that the component is to perform. The state diagrams evolve as the functionality of each component evolves through the derived requirements and design choices. Data flow diagrams are included in the Appendix B to show what information is passed between the different components.

II. BACKGROUND AND HISTORY

A. WHAT IS A KVM SWITCH?

A KVM switch is a device that allows a single keyboard, monitor, and mouse to be shared among multiple computers. The user traditionally selects which computer will receive input and/or output from the peripheral devices on the switch via a push button or rotary knob; however, some switches do allow switching via keyboard shortcuts. A keyboard shortcut is a specific sequence of keystrokes that triggers the switch's internal mechanisms to perform a designated function. In this case, the function is switching from one computer to the next computer.

B. KVM SWITCH TYPES

1. Passive Switch

The first KVM switches were passive mechanical devices. By turning a knob on the front of the device, the user completes the electrical circuit needed for the peripheral devices to interact with the attached computer. The completion of the circuit looks to the computer as if the peripheral devices are being plugged in to the computer directly. Likewise, when the computer is deselected, the computer thinks the peripheral devices were unplugged directly from the computer. The major problem with the passive type of KVM is that many computers will not boot properly if a keyboard is not detected during the boot sequence, and thus, unless the computer is selected via the KVM prior to initiating the computer's boot sequence, the computer will enter into an error state and halt the boot sequence.

Another issue inherit with passive switches is that they rely on contact points to complete the electrical path. The contact points wear down over time and eventually lose the ability to properly connect. Passive KVM switches can still be purchased from electronics stores as they provide basic functionality to meet many users' needs and are relatively inexpensive.

The basic security principles of a passive switch consist of security by separation. The physical connections for each computer are designed in such a way as to ensure only

a single computer can connect to the attached peripheral devices. However, as previously stated, only that computer will be able to detect a mouse and keyboard during the boot cycle.

2. Active Switch

An active switch differs from a passive switch in that it uses electricity to power the electronic circuits within the switch. The electronic circuits allow active switches to combat the main problems that plague passive switches. An active switch can emulate the peripheral devices to allow a computer that is not currently selected to boot properly by registering as both a generic mouse and keyboard or, in some cases as the actual make/model of the attached mouse and keyboard. Also, by using electronic circuits, the contact points can that wear down over time causing the passive switch to fail are no longer an issue.

Most modern day KVM switches are active and support USB interfaces because the user wants a friendlier switch that will not completely disconnect the mouse and keyboard from the non-selected computers. As mentioned above, the user also does not have to worry about which computer is currently selected when booting the systems. Below we describe several existing USB KVMs.

a. Avocent SC4-UAD KVM Switch

The Avocent SC4-UAD KVM Switch was evaluated in 2007 to be National Information Assurance Partnership (NIAP) certified at Evaluation Assurance Level 4[3] (EAL4), meaning that the switch was methodically designed, tested, and reviewed [6]. The reviewer noted that the manufacturing process could lead to compromised switches because of the Flash ROM used in the switch, and because "the isolation tests performed by NSA 172 show that the SC4-UAD does not meet certain specifications for signal separation" [7]. The switch contains some of the properties, as

[3] The National Information Assurance Partnership's *Common Criteria for Information Technology Security Evaluation* [6] states that "EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices which, though rigorous, do not require substantial specialist knowledge, skills, and other resources."

well as lessons learned, that this paper will address as describing requirements for a secure keyboard and mouse switch.

b. BAE Systems Interactive Link

The BAE Systems Interactive Link device combines a KVM switch with a one way diode to allow control of both a low-side computer over a low-side network and a secure window server. The Interactive Link device was evaluated at EAL5[4]; meaning that the device was semi-formally designed and tested. However, the device deviates from the traditional design of a KVM by requiring the use of software on the "high-side" computer that will allow keyboard, mouse and video traffic to be visually displayed from a "low-side" computer on the high-side computer. Essentially, this device creates a secure "remote desktop" experience for a user on a classified network to access server or servers on a network of a lower classification [8].

c. Belkin OmniView Secure KVM

Similar to the Avocent SC4-UAD KVM Switch, the Belkin OmniView Secure KVM was evaluated in 2009 to be NIAP-certified at EAL4, and the switch contains many of the properties, as well as lessons learned, that this paper will address as requirements for our KVM switch [9], [10].

C. USB SPECIFICATION

As the USB specification is the standard for peripheral devices, specifically for keyboards and mice, the switch defined within this paper will utilize USB ports to connect both the peripheral devices and the computers to the switch. Thus, an understanding of the USB specification is necessary to ensure proper controls are put in place to assure high robustness in the design.

The USB specification was initially designed by seven companies looking to make it easier for external devices to connect to computers. Before the introduction of

[4] The National Information Assurance Partnership's *Common Criteria for Information Technology Security Evaluation* [6] states that "EAL5 permits a developer to gain maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering techniques."

USB, external devices were designed with proprietary ports and proprietary protocols. Only the computers designed for the use of those devices could use these devices. The industry saw the need to create a standard that would allow an external device to connect to any computer without the need for expensive add-on cards or complicated software.

The USB bus is host centric, meaning that the host is the "initiator" of all protocol activities. The host sends requests to the USB hub. The hub acknowledges that it received a request but does not respond with the answer to the request from the end point until the host again asks the hub for information. When an endpoint sends data, the last data packet contains all zeroes to tell the host that it is the last packet in that segment.

A hub can provide the ability to increase the number of USB ports available to a host, and multiple hubs can be linked together to form a chain between the host and endpoint. Each hub typically has a single upstream port and multiple downstream ports. A hub's upstream port connects to either a downstream port of the host's USB host controller or a downstream port of another hub's USB function controller. As detailed by the USB specification, only 126 peripheral devices, including hubs, can be attached to a single USB host controller [11].

1. USB Transaction Types

USB transactions are similar to network packets flowing between devices. Each transaction contains packets used to either setup the connection between devices and host or deliver data to the device or the host. The USB specification [11] defines three transaction types:

- IN
- OUT
- Setup

The Message Sequence Chart (MSC) diagrams included in this section are taken from the "Universal Serial Bus Specification Revision 2.0" [11] where more detailed information can be obtained. An MSC is used "to describe the interaction between a number of independent message-passing instances [12]."

a. IN Transaction Type

The IN transaction type allows data that originates from the endpoint to be delivered to the host. The host must first send a request to the endpoint requesting data from the endpoint starting with a start split (SSPLIT) packet and an IN packet. The hub then sends an acknowledgement (ACK) to the host upon receiving the request but prior to forwarding the request on to the endpoint. The endpoint responds to the host's request with data packets or a Negative Acknowledgement (NAK) packet if there is no data available to send to the host.

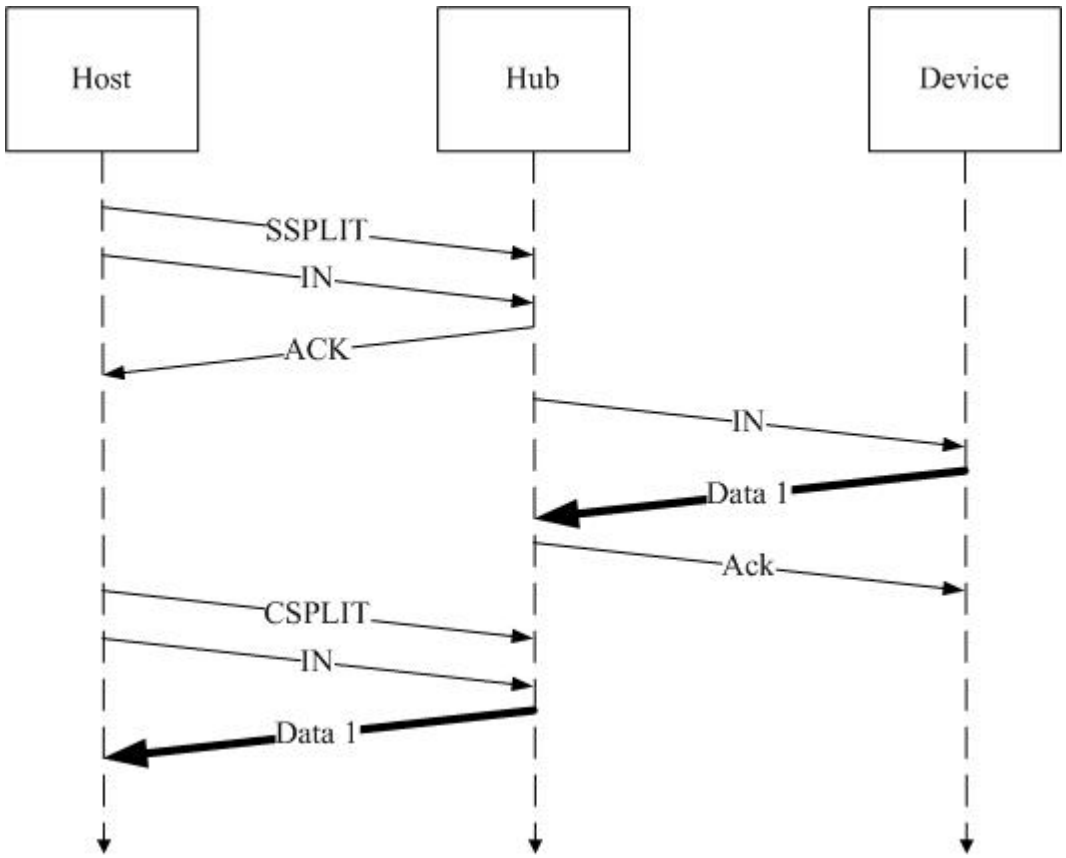

Figure 3. USB IN MSC. From [11]

As with any sequence based protocol, data transfer errors can and do occur. If the error occurs between the host and the hub, the host will automatically resend the request up to a maximum of three attempts if it doesn't receive a response after a set amount of time. The hub will pass along the request to the endpoint once received. If, however, an error occurs after the endpoint has acknowledged the host's request and

the host is now asking to complete the split (CSPLIT) packet, the hub will respond with a no response yet (NYET) packet telling the host that the request was accepted but it has not yet received any information from the endpoint. The host will then know to resend the CSPLIT command at a later time.

b. OUT Transaction Type

The OUT transaction type permits data to be transmitted from the host to a device. The host sends the data to the device, and the device responds with an ACK packet if successful or a NAK packet if an error occurred during the transaction.

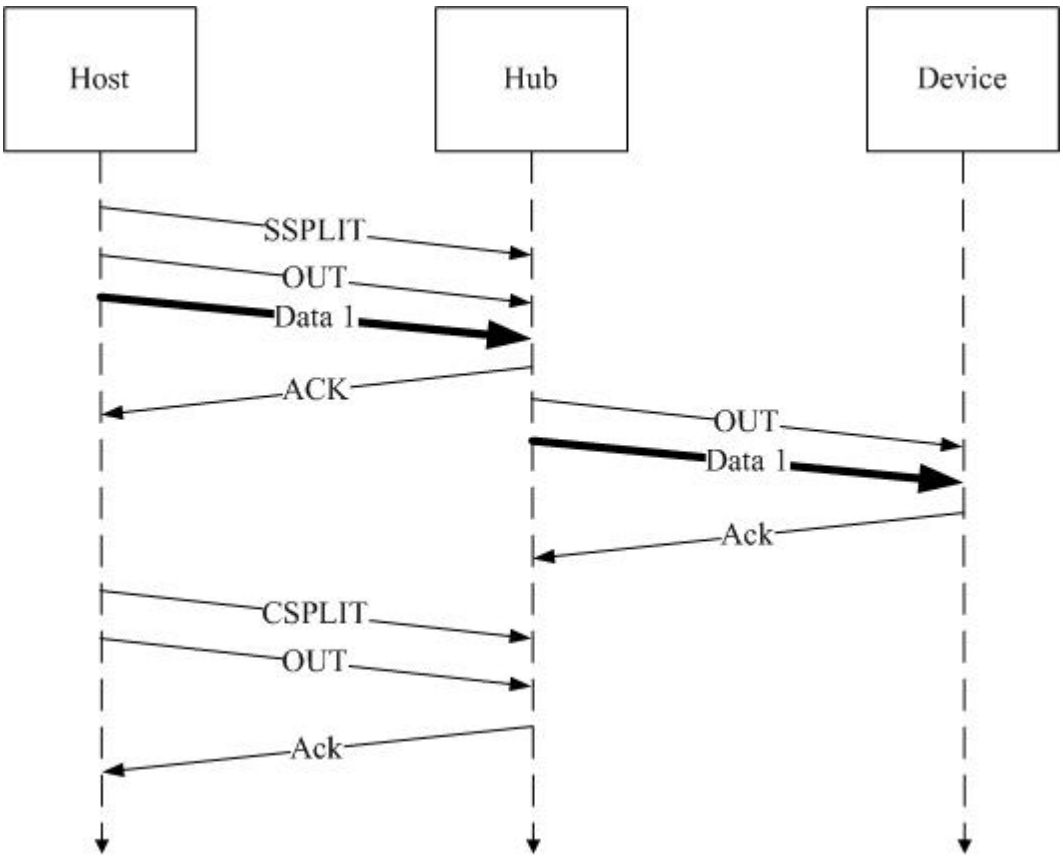

Figure 4. USB OUT MSC. From [11]

c. Setup Transaction Type

The Setup transaction type is similar to the OUT transaction type, except that a device must accept and respond to a Setup transaction packet.

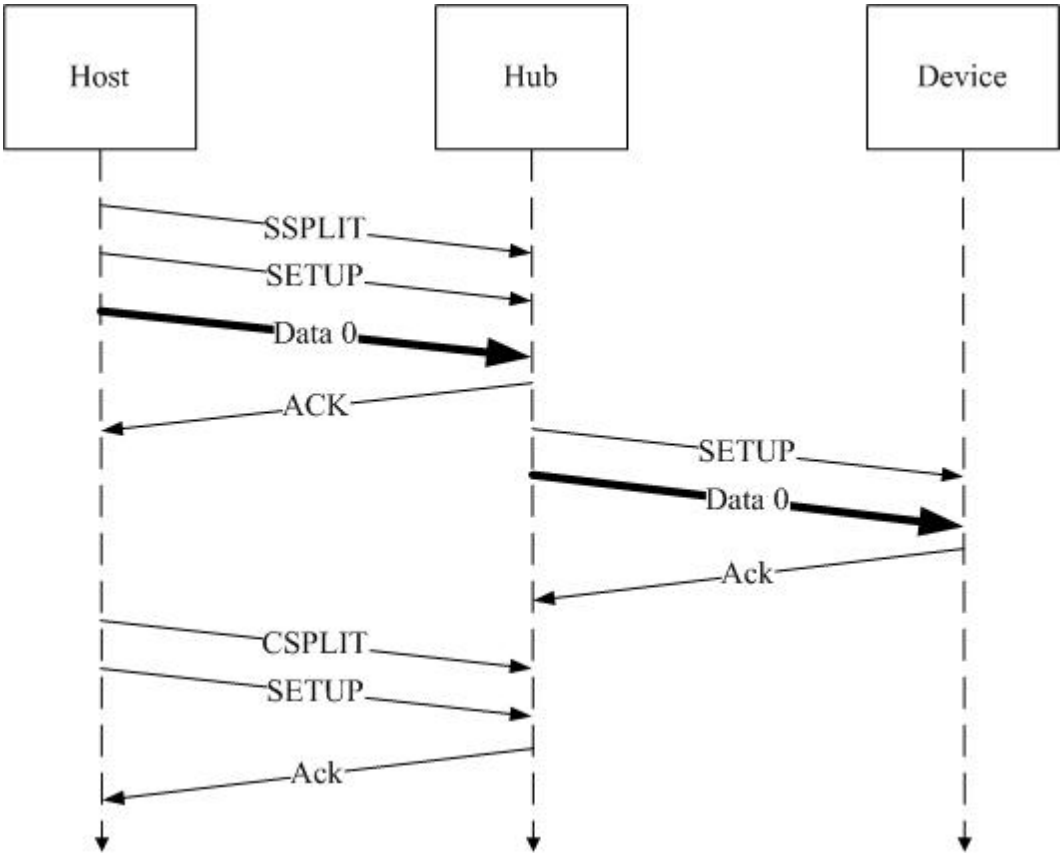

Figure 5. USB Setup MSC. From [11]

THIS PAGE INTENTIONALLY LEFT BLANK

III. EXTERNAL INTERFACES

A typical KVM switch defines the external interfaces to be ports for connecting input devices such as keyboards and mice, port or ports for connecting each computer, switching devices such as buttons, indicators to show which computers are connected, indicators to show which computer is currently selected, and a port to connect to a power supply. Figure 6 and Figure 7 depict a generic 2-port KVM. The keyboard and mouse ports on the front of the switch are part of the Input Peripheral Port Group (IPPG). On the back of the switch, the port used to connect to the computer is designated as the Computer Peripheral Port (CPP). A 2-port KVM has two distinct CPPs with a selection device used to switch between the two CPPs.

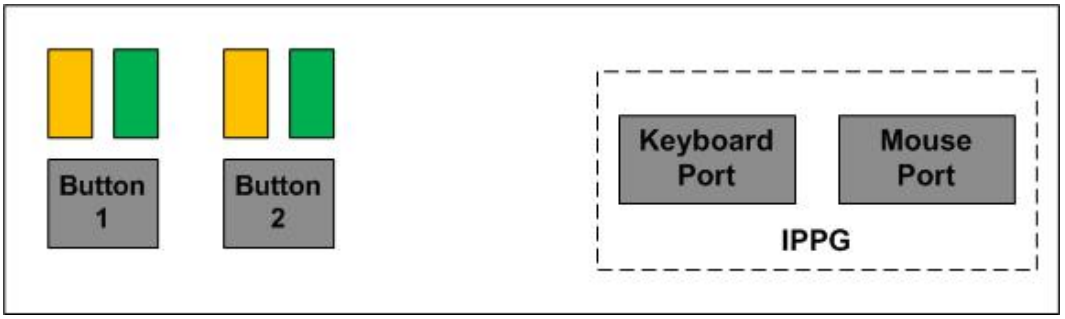

Figure 6. 2-port KVM: Front

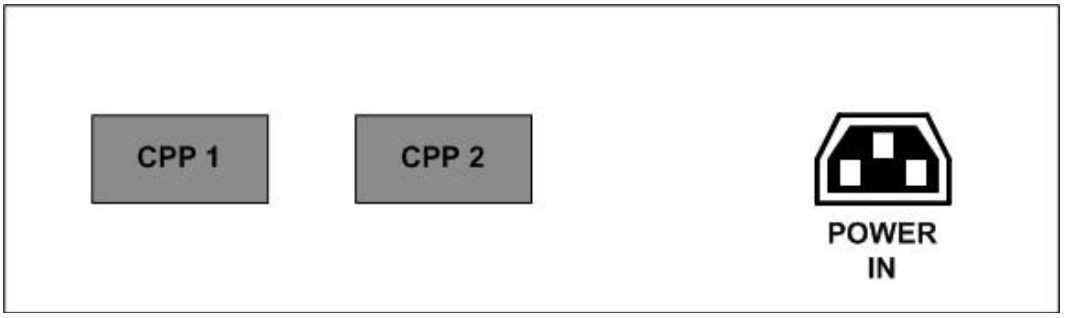

Figure 7. 2-port KVM: Back

THIS PAGE INTENTIONALLY LEFT BLANK

IV. USE CASE SCENARIOS

The following use case scenarios will help to define the requirements of the system. Some of the cases will describe the flow of a user interacting with the system with a MSC depicting the flow of information, but other cases will simply provide a description of the functionality provided by the switch.

A use case scenario describes the interaction between a user and a system. Each scenario describes inputs into the system, either external or internal, and their expected outcomes. An external input can be described as any action performed that originates from the outside of the physical system. Likewise, an internal input describes any action performed by the system that originates from within the physical system. An internal input is usually the result of or an output from an external input.

A. INITIAL SETUP

In this section, we describe how the user interacts with our proposed KVM during the initial setup phase by using the written guidance provided with a specific switch. The installer connects the user's keyboard and mouse cables to the IPPG on the switch. The installer connects each of the user's computers (1 to n) to the corresponding CPP, labeled 1 to n, on the switch. The installer will use two cables, one for the mouse and one for the keyboard. The installer labels the buttons (1 to n) on the front panel of the switch with the identity of the corresponding computer (e.g., Computer 1/Computer 2). Note: This cabling may be connected in any order, but it will be assumed that the user will correctly connect the cables. Lastly, the installer connects the switch's power cable to the power outlet.

B. POWER-ON CYCLE

The system can be described as having two distinct power-on cycles. Either the switch is powered on prior to one or more of the computers connected to it, or one or more of the computers are powered on before the switch. Below we describe each of these.

1. Power-On Switch before Computers

Following the Initial Setup phase, the user turns the switch on by plugging in the switch's power cable. We assume that there is no separate "on-off" switch. During the switch's boot cycle, the switch will initialize and set up the pathway between the IPPG and the attached keyboard and mouse. At some point in time following the initialization of the switch's power on cycle, the user will power on any number of the computers (1 to n). During each computer's boot cycle, the computer will communicate with the switch to initialize and set up the pathway between the computer and the CPP over the attached cables. The computer(s) will see the switch ports as a generic hub with a generic mouse and keyboard attached to it. The switch will provide a visual indication to the user once the switch detects the computer(s) are connected and powered on. The sequence of operations is described in the message sequence chart show in Figure 8.

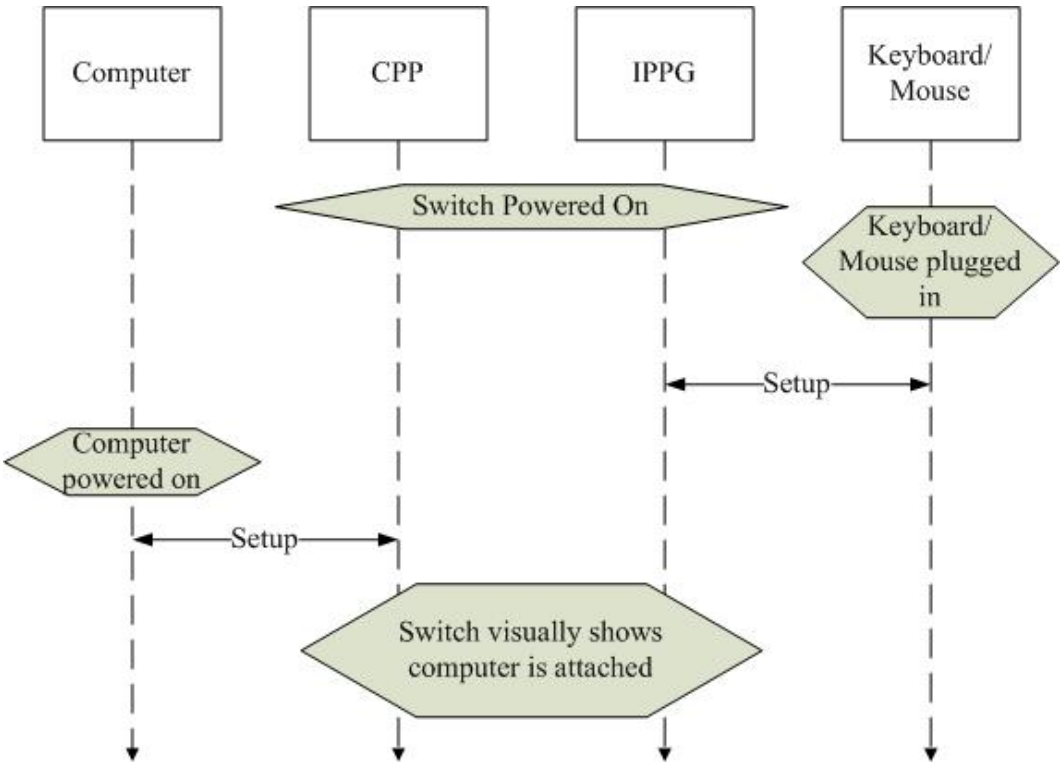

Figure 8. Message Sequence Chart: Switch Powered On before Computer(s)

2. Power-On Computers before Switch

In this scenario, the user does an "Initial Setup" before plugging in the KVM. Following the "Initial Setup" phase, the user then powers on any number of the computers (1 to n). Since the KVM is unpowered, no physical keyboard or mouse is visible to the computer. So that the computer(s) can boot properly, the switch's CPP emulates a physical keyboard and mouse. The computer "thinks" it is creating a pathway between itself and a physical keyboard and mouse through the switch's CPP. At some point the user then powers on the switch by plugging in the switch's power cable. During the switch's boot cycle, the switch tears down the temporary pathway between the computer(s) and the emulated keyboard and mouse. The switch then allows a new pathway between the computer(s) and CPP(s) to be setup. Also during the switch's boot cycle, the switch will initialize and set up the pathway with the attached keyboard and mouse. Upon completion of the computer's boot cycle, the switch will visually show which computers are attached and powered on. By default, no computer will be selected as the default pathway and thus the switch will wait for the user to select a computer for use before setting up the pathway between the attached keyboard and mouse and any of the computers. This process is shown in the message sequence chart, Figure 9.

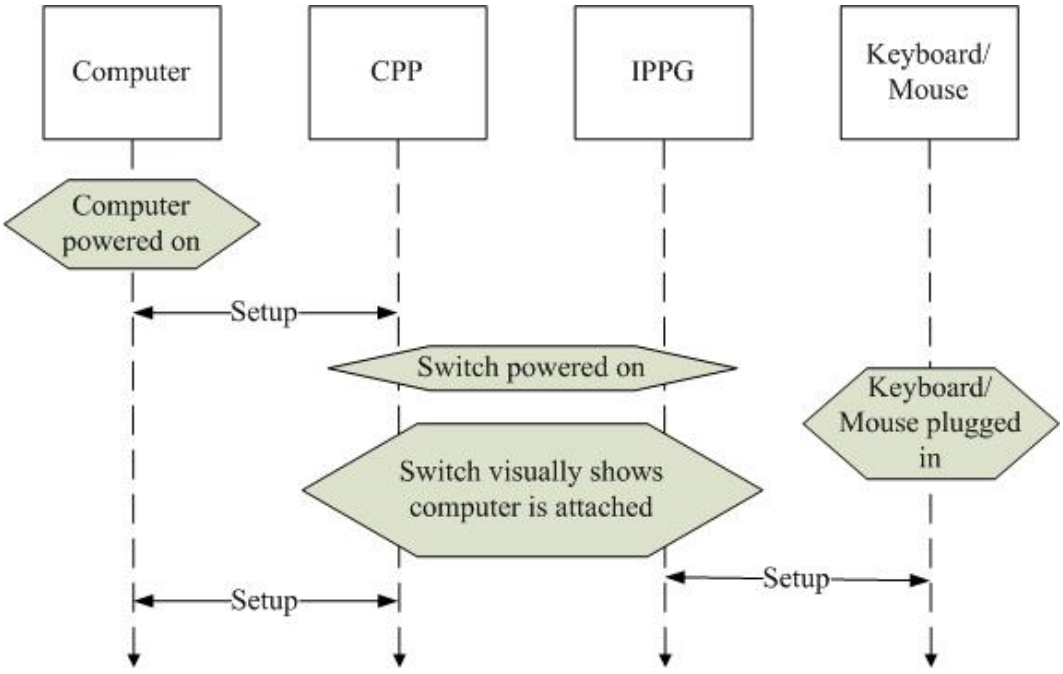

Figure 9. Message Sequence Chart: Computer(s) Powered On before Switch

C. STANDARD OPERATION

During “Standard Operation” we assume that one or more computers, a keyboard, and (optionally) a mouse are connected properly to the KVM and that the switch and computers have been powered on using one of two methods described above. The user presses one of the buttons (1 to n) on the switch that correlates to an attached computer. The goal is to establish a connection between the attached computer and the keyboard and mouse. The switch will visually show which button is currently selected while the data pathway is setup. The switch will initialize and setup the data pathways between the attached keyboard and mouse and the selected computer. As soon as the pathway is correctly set up, the switch will visually show to the user that the user is now able to send input to the attached computer via the keyboard and mouse. At some point, the user will press a different button on the switch. The switch will visually show which button is currently selected. The switch will tear down the pathway from the Keyboard/Mouse and the previously selected computer, ensuring no residual data can flow to the newly selected computer. The switch then proceeds to setup the pathway between the Keyboard/Mouse and the newly selected computer. The switch will visually show to the user that the user is now able to send input to the attached computer via the keyboard and mouse. The sequence of operations is described in the message sequence chart show in Figure 10.

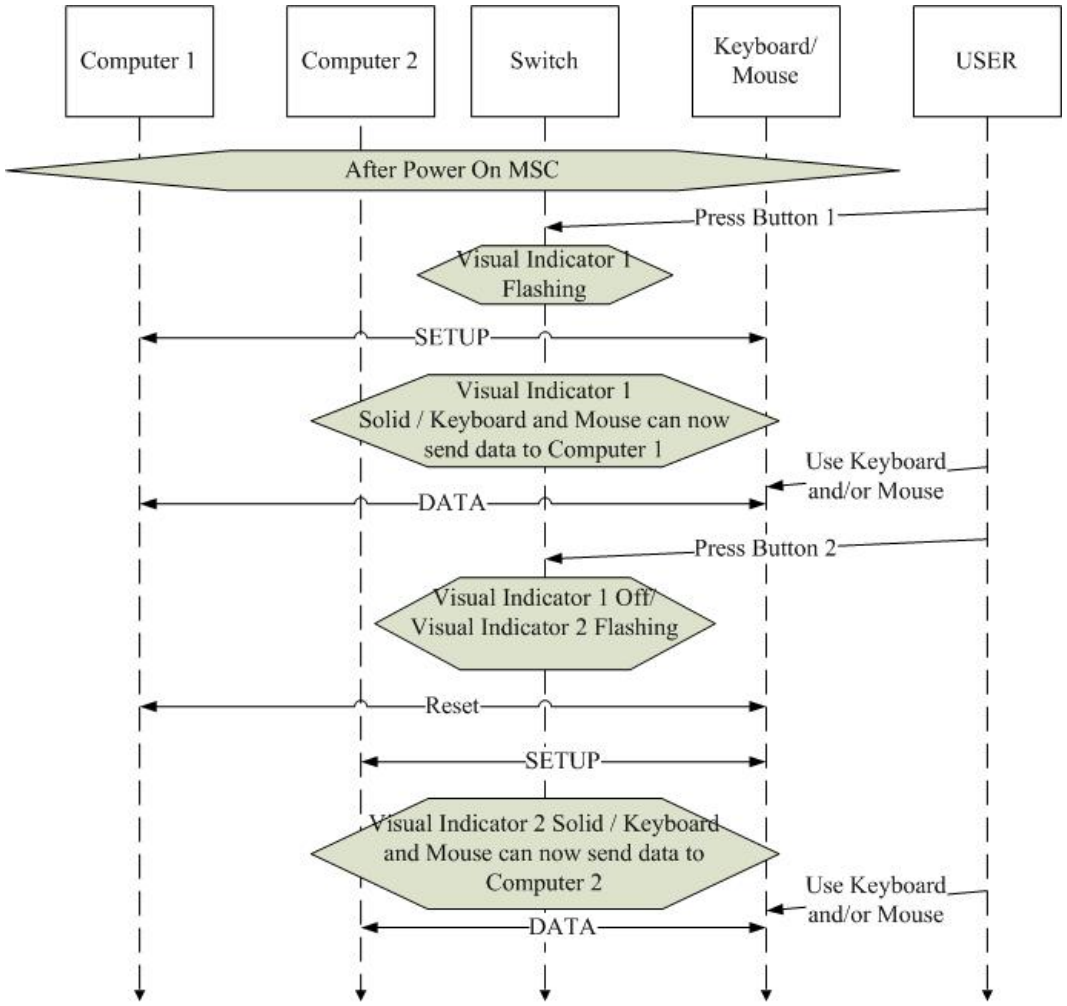

Figure 10. Message Sequence Chart: Standard Operation

D. THE CURRENTLY SELECTED COMPUTER IS SELECTED AGAIN BY THE USER

At some point during the standard operation scenario, the user may reselect the currently selected computer. The switch shall disregard the user's request as no action is required.

E. KEYBOARD AND/OR MOUSE IS PLUGGED-IN OR UNPLUGGED AFTER THE SWITCH IS POWERED ON

If the attached keyboard and/or mouse disconnects from the switch, the switch will tear down the connection between the keyboard and mouse. If a computer is selected

while the keyboard and/or mouse are disconnected, then the switch will setup the generic keyboard and mouse emulation. The user is then free to reconnect the same or different keyboard/mouse to the switch. Once connected, the switch will initialize and setup the connection with the keyboard and/or mouse. If a computer is currently selected, the switch will setup the pathway between the Keyboard/Mouse and the currently selected computer.

F. CURRENTLY SELECTED COMPUTER IS DISCONNECTED FROM THE KVM

If an attached computer, which is the currently selected computer, is powered off or otherwise disconnects from the switch, the switch will tear down the pathway from the Keyboard/Mouse and the currently selected computer**.** The switch will no longer visually show the currently selected computer as a viable option and will reset the ports in anticipation of a different computer connecting to them. The user will need to select another computer for use.

G. ADDITIONAL COMPUTER(S) ARE CONNECTED TO THE SWITCH AFTER A PREVIOUSLY ATTACHED COMPUTER IS SELECTED

A computer is powered on and attached to the switch after the switch is powered on and after a separate computer is powered on, attached to the switch, and selected via the switch's front panel button. The newly attached computer will detect the switch as a powered hub with a generic keyboard and mouse attached to the hub. Once the switch detects the computer is connected and powered on, the switch will visually show that the computer is attached and powered on.

H. COMPUTER NOT CURRENTLY SELECTED IS DISCONNNECTED FROM THE SWITCH

If an attached computer that is not currently selected is powered off or the computer's cables are disconnected, the switch will tear down the connection between the computer's cables and the switch ports and reset those ports in anticipation of a possibly different computer connecting to them. The switch will also no longer visually represent the disconnected computer as a viable choice.

I. THE SWITCH LOSES POWER

If the switch loses power without the user unplugging the power cable, the switch will not gracefully shutdown. All pathways between the attached computers and the switch as well as the switch and the attached keyboard and mouse will be terminated. Any computer attached to the switch that is currently powered on will be responsible for providing power to its respective CPP. The switch's CPP will use the supplied power over the attached cables to emulate an attached Keyboard/Mouse. The switch will not visually represent which computers are attached and powered on.

J. FAIL SECURE

In the unlikely event that some part of the switch malfunctions or otherwise becomes inoperable, the switch shall fail secure. The secure failure state shall be defined as "no data shall be allowed to flow through the switch." To mitigate the potential of a component failing, the switch shall be designed and built using reliable hardware.

THIS PAGE INTENTIONALLY LEFT BLANK

V. MISUSE CASE SCENARIOS

Use case scenarios are helpful to describe how the user is intended to utilize the switch in day-to-day operations, however, certain situations should be considered to ensure the integrity of the switch is maintained. These misuse case scenarios, describe when the user, either intentionally or unintentionally, attempts to use the switch in a manner that is prohibited by security posture of the switch. The switch should be able to detect these events and prevent them from occurring.

A. UNAUTHORIZED DEVICE IS PLUGGED INTO THE KEYBOARD AND/OR MOUSE PORT

A user may try to plug in a non-approved device into the keyboard and/or mouse port during standard operation. The switch will detect the unapproved device and disable the port. The switch will re-enable the disabled port upon detection of a new, approved device connecting to the port.

B. THE CURRENTLY SELECTED COMPUTER SENDS DATA PACKETS INTENDED FOR A NON-HUMAN INTERFACE DEVICE (HID)

The currently selected computer, either through error or malicious activity, may try to send data packets that a HID device should not receive, such as data intended for a storage device. The switch shall drop these packets before they are allowed to reach the attached keyboard and/or mouse.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. REQUIREMENTS AND ARCHITECTURAL REFINEMENTS

Requirements are essential to creating an accurate and complete design. A design without requirements has no purpose and can never be considered complete. Requirements provide test accuracy by which a system or design can be verified for correctness. There are many types of requirements that can be assigned to a system from basic functionality to specific architecture requirements.

The basic or functional requirements define the core functionality of the system. Engineers and architects take the core requirements and start to implement them within a physical design. During the design process, additional requirements or derived requirements may surface based on architectural decisions or constraints.[5]

In this section, each requirement is defined and then discussed. State diagrams are included and are expounded upon as requirements mature or additional requirements are added to the components of the switch. From the core requirements derived requirements and design choices will be introduced with diagrams of the switch components, the components' state diagrams, and tables describing the variables used in the state diagrams to help guide the reader.

A. CORE REQUIREMENTS

The main purpose of the switch is to provide the ability for a number of computers to be connected to a single keyboard and single mouse. However, only one of the attached computers should be allowed to communicate with the keyboard and mouse at any given point in time.

C-1. The switch shall enable communication between a keyboard and a mouse and at most one computer connected to the switch at a given point in time.

[5] Core requirements, derived requirements, and design choices are denoted by "C-xx", "D-xx", and "DC-xx," respectively.

Now that the base functionality of the switch has been defined, the security posture should be established. This will ensure that data flows are well defined and all external interfaces are secured.

C-2. Prevent communication via the switch between attached computers.

The switch is not intended to be a data transfer device and should not allow any data to pass between computers either directly or indirectly. If a user should wish to move data between the attached computers, the user should find another means to do so, such as over a network, through a Cross Domain Solution, or via external media (i.e., CD/DVD or external hard disk drive).

Devices, such as CAC Readers, may be required to access a particular system or service that is provided by one or more of the attached computers. Other devices, such as hubs and splitters, allow the number of defined ports on the switch to be increased so that more peripheral devices can be attached to the switch. However, devices such as these will not be allowed to communicate through the switch.

C-3. The switch shall not allow communications with peripheral devices connected to the IPPG other than a keyboard on the keyboard port and a mouse on the mouse port.

Along with the core functionality and security requirements, the switch will need to provide a way for a user to select which computer is permitted to communicate with the keyboard and mouse. This should give the user confidence that the keyboard and mouse will be interacting with the correct computer.

C-4. The switch shall allow the user to select the computer to be paired with the attached keyboard and mouse.

Once the user has selected a computer with which to interact, the user may, at some point, need to be reminded which computer is actively paired with the mouse and keyboard. This indicator will also be useful for a new user accessing the switch and computers to instantly know which of the computers, if any, are actively paired with the keyboard and mouse.

C-5. The switch shall, indicate which computer, if any, is paired with the attached keyboard and mouse.

The switch needs to operate transparently and allow the connected external computer systems to operate normally. At startup, most computer basic input/output systems (BIOS) run diagnostic checks to determine if a keyboard is attached and if the keyboard is functioning properly. If the computer's BIOS does not detect a keyboard attached to the system during startup, the system will normally halt any further operations. The switch will function in such a way as to prevent this error condition from occurring while the switch is in either state, powered on or powered off.

C-6. The computers connected to the switch shall be able to boot normally.

B. INITIAL DERIVED REQUIREMENTS AND DESIGN CHOICES

Derived requirements are "requirements that are implied or transformed from higher-level requirement" [13]. Design choices reflect the outcome of the analysis of alternative design implementations. The development process typically alternates between refinement of requirements (core and derived) and refinement of the design (design choices).

D-1. The switch shall be powered.

As the switch provides an indicator that dissipates power, the first and second laws of thermodynamics dictate that the switch consumes power from either an internal or external source.

DC-1. Standard wall power will be used to power the device.

In addition to providing an indicator, the switch will likely incorporate other power consuming electronics. For engineering convenience the switch will utilize standard wall power, such as can be found in a typical computing environment, to provide power while the device is powered on. This is in lieu of alternative power sources such as battery or solar power that have limited life or availability. The exact power standard will be determined by the electrical engineer who implements the design based on the circuitry used and the country's or countries' electrical standards to be supported.

DC-2. Conventional COTS IT and electronics shall be utilized.

To both save time and money, the device will consist primarily of commercial off the shelf (COTS) information technology (IT) and electronics to the extent that the security posture of the switch can be verified. Custom components may be needed to combine the standard technology into a functioning device, but their use will be minimized where possible.

D-2. Core requirements C1, C-3, C-4, and C-5 are applicable only when the switch is powered on.

As conventional active electronics will be used in the switch, communication through the switch is only possible when it is powered on. Being able to select a computer and view the selection is only useful when communication is possible. Note that core requirements C-2 and C-6 are applicable whether the switch is powered on or off.

D-3. The switch shall not allow communication between the keyboard and mouse and any computer when the switch is powered off.

Derived requirement D-2 only addresses what occurs when the power is on. The implied behavior of no information flow when the switch is off is likely implemented in the default behavior of the switch.

D-4. As viewed externally, the switch shall have a finite set of atomic state transitions.

An atomic state transition is defined as a transition that does not require any intermediary steps to change states.

D-5. The switch shall visually present a finite set of internal states of the switch.

The set of internal states that the switch can visually present to the user is the status of the attached computer(s), the status of the connected keyboard and mouse, and the status of the currently paired computer.

DC-3. The switch will support at least two computers.

The use of two computers is the least costly case.

DC-4. The switch will present the internal states identifying which computer is paired with the attached keyboard and mouse through the switch.

With the switch powered on, the user shall be provided a means to quickly determine which CPP, if any, is currently selected. By providing a quick reference for the user, the switch can better manage the user's expectation of which computer is currently able to interact with the keyboard and mouse. If the switch is not powered on, the user will be unable to use the attached keyboard and/or mouse to interact with any attached computer. Therefore, when powered off, the switch does not need to provide a means for the user to determine which computer is currently selected.

Figure 11 depicts the finite set of atomic states for a two computer KVM that the user will see before and after pressing a "user selection device." The initial state assumes the switch is powered on, and one or more computers are attached and powered on.

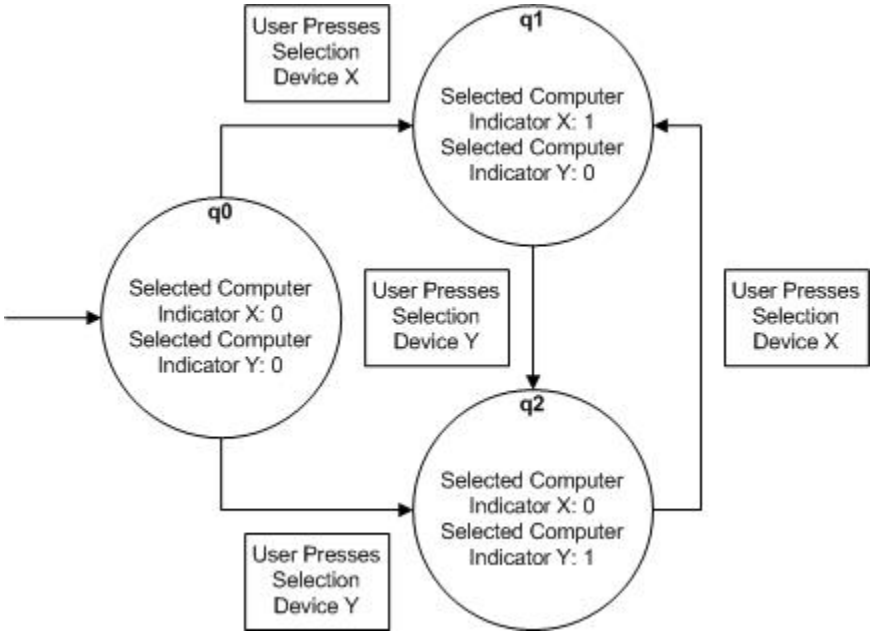

Figure 11. Basic Switch States as Seen from the User's Perspective

Each state in Figure 11 is defined by the variables in Table 1:

Variable Name	Values	Description
Selected Computer Indicator (X or Y)	0	The visual indicator defined by the Selected Computer Indicator (X or Y) is off.
Selected Computer Indicator (X or Y)	1	The visual indicator defined by the Selected Computer Indicator (X or Y) is on.

Table 1. Basic Switch States' Variables, Values, and Descriptions

DC-5. The main components of the switch will consist of an IPPG, a finite number of CPPs, a switch module, and a user interface (see Figure 12).

- **IPPG**. The IPPG provides connection ports on the switch that allow a single keyboard to be plugged into the keyboard port and a single mouse to be plugged into the mouse port.

- **CPP**. Each CPP provides a connection port on the switch that allows one computer to be plugged in at a time.

- **Core Switch Module (CSM)**. The CSM controls the process of switching from one CPP to another, as well as, ensures the security posture of the device is maintained. The CSM communicates with the IPPG, the CPPs, and the user interface. The module contains a processor and the memory needed to properly support the functions of the switch.

- **User Interface**. The user interface provides the means by which the user selects the desired computer and determines which computer, if any, is currently selected. The user interface contains user selection devices and selected computer indicators.

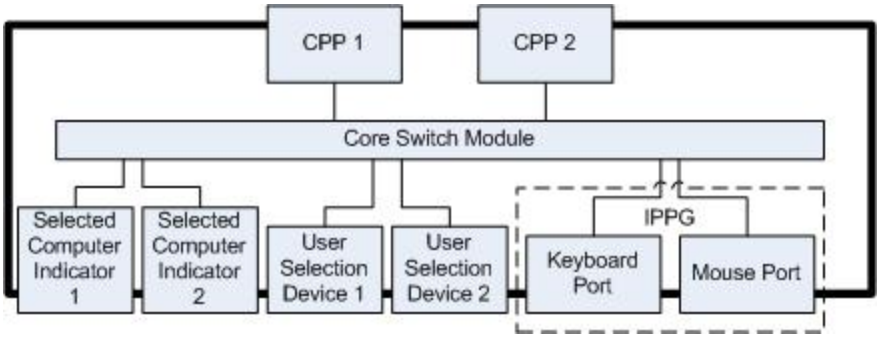

Figure 12. Initial Design Diagram

D-6. The switch shall only pair with an active computer connected to a CPP.

Only active connected computers can provide viable communications. A non-connected computer or computer that is powered off will be unable to provide the required input(s) into the switch.

DC-6. The switch will indicate which computers are available.

The switch will provide an indicator for each computer displaying its availability (i.e., connected and active). This indicates to the user which computer selection choices will be allowed. Thus the User Interface component of the switch design should include an "Available Computer Indicator" (see Figure 13).

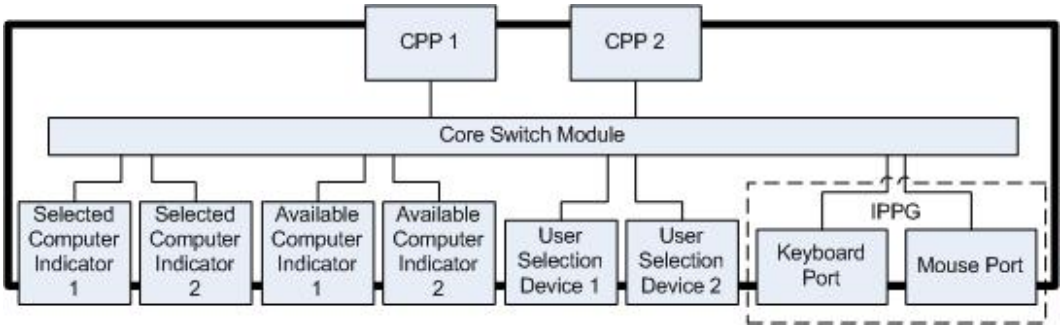

Figure 13. Initial Design with Available Computer Indicators

Figure 14 depicts the finite set of states of the switch before and after connecting a computer to the switch. Each state contains both an internal state and an external state.

The internal state, Computer X Available, allows the switch to track which CPP has an active computer connected while the external state, Available Computer Indicator X, allows the switch to visually represent to the user which CPP is connected to an active computer. The initial state assumes the switch is powered on and one or more of the computers is powered on.

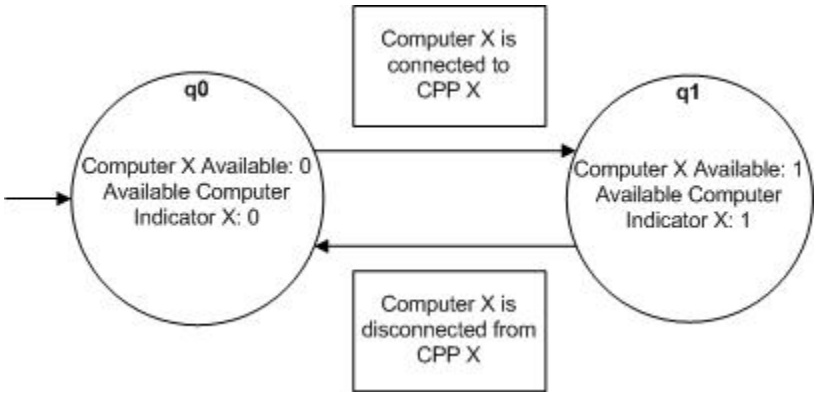

Figure 14. CPP–Basic State Machine

Each state in Figure 14 is defined by the variables in Table 2:

Variable Name	Values	Description
Computer X Available	0	Computer X is either not connected to CPP X or it is connected to CPP X but not powered on.
Computer X Available	1	Computer X is connected to CPP X and is powered on.
Available Computer Indicator X	0	Computer X is either not connected to CPP X or it is connected to CPP X but not powered on.
Available Computer Indicator X	1	Computer X is connected to CPP X and is powered on.

Table 2. CPP's Basic Variables, Values, and Descriptions

D-7. At no point in time shall two CPPs be allowed to communicate through the core switching module.

As the switch transitions from state to state based on both internal and external state changes, no data shall be allowed to pass between CPPs. Figure 11 represents the basic states showing that at no time will the switch be able to enter a state in which both CPPs are actively connected to the peripheral Keyboard and Mouse. This implies that two CPPs will neither be allowed to communicate with each other nor simultaneously communicate with the IPPG. The requirement ensures that the fundamental security principle of the switch is properly supported.

D-8. The switch shall protect the integrity of the communications between computer selection devices, the selected computer indicators, the CPPs, and the CSM itself.

The computer pairing selected by the user, the computer pairing indicated to the user, and the actual computer pairing must be identical. The integrity of the security enforcing behavior of the selection operation must be maintained.

DC-7. Protective Design.

The switch will use a protective enclosure with anti-tamper features to prevent unauthorized access to security components and their communication paths. The internal design will implement the security principles of domain separation, process isolation, resource encapsulation, modularity, simplicity, least privilege, secure initialization, safe failure, and trusted recovery.

C. DESIGN CHOICES – EXTERNAL INTERFACES

The switch's external interfaces are the points of the device where the user will either plug-in a device or provide input to the switch. As previously defined, the CPPs and IPPG will consist of ports that will enable a keyboard and mouse attached to the IPPG to communicate with a computer attached to one of the CPPs. The ports could be defined by any number of standard interfaces in use today such as IEEE 1394 or FireWire, PS/2, serial, or USB. Any of these defined protocols would allow a mouse and keyboard to interact with a computer. However, the IEEE 1394 protocol is overkill, with

up to 400Mbit/s, for the amount of data that would need to be transferred between the keyboard and mouse and computer. Both PS/2 and serial interfaces have been replaced in most modern computing devices by the USB interface. Thus the switch shall use USB as the defined interface and protocol.

The USB standard defines a protocol that will allow the user the flexibility to choose from the large number of standard USB keyboards and mice available on the market today. The use of the USB standard will also help to keep the cost of designing and building the switch lower. If a non-standard protocol or interface is implemented in the switch, the designer would need to potentially provide a custom keyboard and mouse along with cables to connect the switch to the computers. If the keyboard, mouse, or any of the cables were to fail, the user would be forced to contact the switch supplier. The user would possibly experience unacceptable downtime if the supplier is unable to repair or replace the malfunctioning equipment in a timely manner.

DC-8. The external ports to the switch will be defined by the USB standard.

In order to properly support the USB ports, a separate USB function controller will be paired with each port group, one for the IPPG and one for each CPP. The controller paired with each CPP will be responsible for correctly communicating with the upstream USB host controller in the attached computer. Likewise, the controller connected to the IPPG will be responsible for correctly communicating with the currently selected upstream CPP.

DC-9. Each CPP and the IPPG will be coupled with a USB Controller.

The switch will also need to provide an interface by which the user can select which computer is allowed to communicate with the keyboard and mouse. A collection of physical buttons provides the user the means to choose between the different computers attached to the switch. The buttons will also provide tactile feedback so that the user knows the button press has been registered by the switch.

DC-10. The switch will utilize physical buttons for computer selection.

An interface is defined as the place at which independent and often unrelated systems meet and act on or communicate with each other.[6] An interface can provide input(s), output(s), or both between the systems that connect at the interface. The buttons on the front of the switch allow the user to provide input into the CSM. In return, each button shall be paired with a green LED light that will illuminate when the switch detects a button press. The light shall serve as a visual representation of the user's current CPP selection.

DC-11. The switch will utilize a GREEN LED light to visually represent the currently selected CPP.

The user can use the light as a visual reminder of which CPP is currently selected; however, the user may not know which CPP is currently connected to an active computer. The switch shall provide a second visual indicator, an amber LED light, to represent an active CPP, which is defined by an attached computer that is powered on.

DC-12. The switch will utilize an AMBER LED light to visually represent an active CPP.

After pressing a button to select a CPP, the user can expect a slight delay while the switch either transitions between CPPs or initially configures the path between the IPPG and the selected CPP. To ensure a positive user experience, the GREEN LED shall not fully illuminate until the path between the selected CPP and the IPPG is fully setup.

DC-13. The GREEN LED will blink while the path between a CPP and the IPPG is being setup.

Table 3 summarizes the LED light colors and functionality.

[6] See http://www.merriam-webster.com/dictionary/interface.

LED Color	Functional Description
Amber	A computer is attached to the CPP and powered on.
Green	Data can flow from the IPPG to the CPP
Green Flashing	The CPP is selected, but the device will not allow data to flow from the IPPG to the CPP yet.

Table 3. LED Colors and Functional Descriptions

With the external interfaces clearly defined, an accurate drawing (see Figure 15) of the currently defined switch components is below.

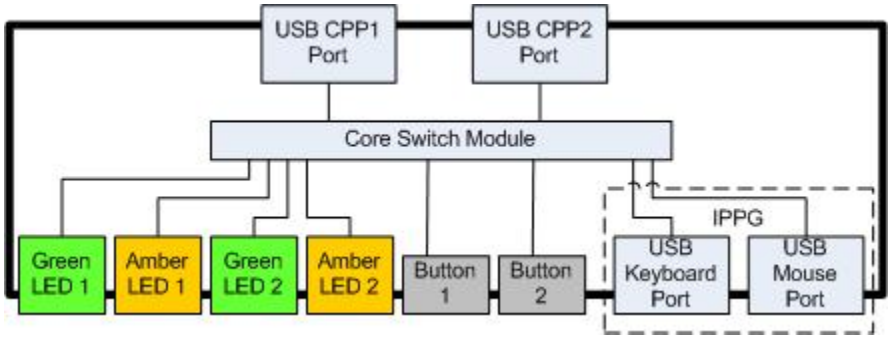

Figure 15. Switch Including Initial Design Choices

D. DESIGN CHOICES – SECURITY SUPPORTING FUNCTIONS

Security supporting functions are defined as functions that do not directly enforce the core security principles but assist in providing assurance that the switch remains in a secure state. The switch will need to provide assurance that the correct CPP corresponding to the button pressed by the user is communicating with the peripheral devices attached to the IPPG. To provide this assurance, the buttons will need to directly communicate with and only with the CSM.

DC-14. The buttons on the front panel will only communicate with the CSM.

As each green LED light is paired directly with a single button to visually represent the user's selection, the green LED lights should be controlled by the CSM as

well. The switch will utilize the green LED lights to provide the user assurance that the path between the IPPG peripheral device(s) and the selected CPP is trustworthy and functioning correctly.

DC-15. The green LED lights will be controlled by the same secure processor as the buttons.

The attached computers will need access to a keyboard to prevent boot errors. However, when the switch is powered off or powered on but the CPP is not currently selected, the computer will not be able to communicate with the keyboard attached to the IPPG. Thus, the switch shall couple each CPP with an emulation module (EM) that will be responsible for mimicking a generic keyboard. This will allow any computer to boot or reboot at any time while attached to the switch without fear of hanging during the boot process due to a keyboard detection error.

Emulation involves allowing a computer attached to a CPP to logically think that a physical keyboard is plugged in and available via the USB cable even though a physical keyboard is not connected.

DC-16. Each CPP will be coupled with an EM designed to emulate USB keyboard functionality to ensure any attached computer can properly boot.

As each EM will need to operate while the switch is powered off, each EM will need to draw power via the USB cable connected to the attached computer. Once the switch is powered on, the emulator shall draw its power from the switch power supply.

DC-17. Each EM will be powered via USB from the attached computer while the switch is powered off.

E. DESIGN CHOICES – SECURITY ENFORCING FUNCTIONS

Security enforcing functions are defined as functions that directly affect or determine the security posture of the device. The components of the switch that have the responsibility of ensuring the switch remains in a secure state comprise the trusted computing base (TCB). The TCB consists of the hardware, software, and firmware that is responsible for enforcing the security policy [13].

The USB protocol defines the subclass field to report if an HID supports a boot interface. The IPPG can read this field during the initial setup phase that occurs when the switch is powered on and the IPPG detects a USB peripheral on the Keyboard and/or Mouse port. Once the IPPG determines the value of the subclass field, the IPPG will either allow further communications, if the subclass equals 1, or disallow further communication to or from the non-bootable HID(s) connected to the front port(s) of the switch.

DC-18. The IPPG will be coupled with a secure module to ensure that only USB packets originating from a HID are transmitted through the switch.

The IPPG can further use the USB protocol's bInterfaceProtocol field to determine if the bootable HID is a keyboard, a mouse, or some other HID. The HID should set the field to "1" for a keyboard, "2" for a mouse, or "0" for other during the initial USB Setup process with the IPPG. Thus, assuming the HID is trusted to set this field correctly, the IPPG will be able to block communications from a non-HID, a mouse attached to the keyboard port, and/or a keyboard attached to the mouse port as required by the core requirement C-3. In addition, the IPPG will send a signal to the CSM indicating the status of "Keyboard Available." The basic states for the IPPG can be seen in Figure 16.

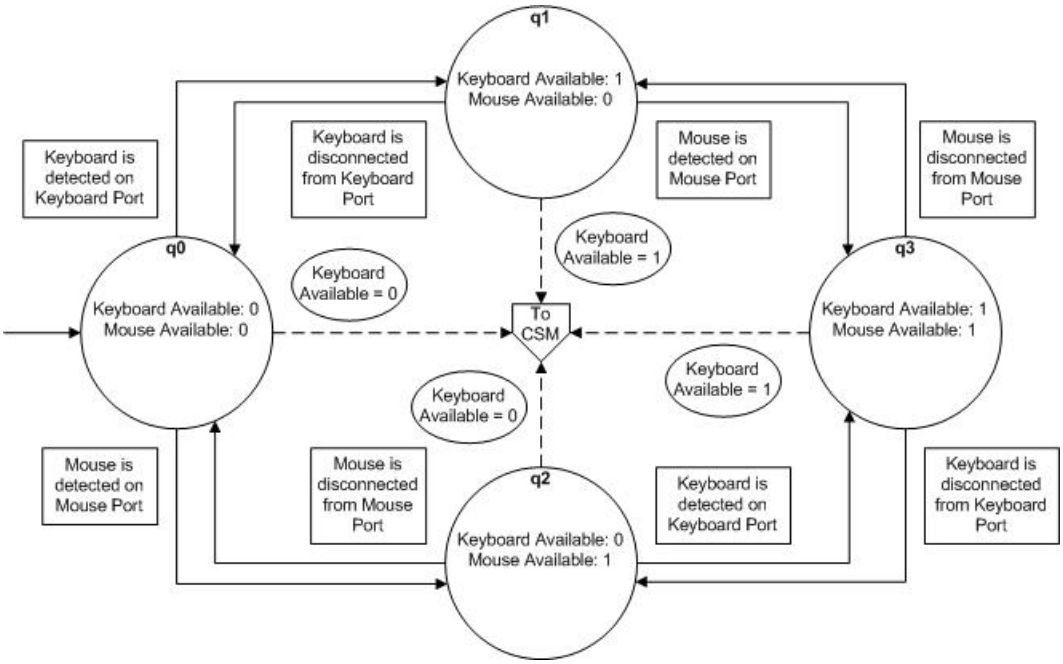

Figure 16. IPPG–Basic State Machine

Figure 17 then illustrates the error state that the switch will transition to when a non-HID device is detected on the Keyboard Port, the Mouse Port, or both. The same error state will also be utilized for when a keyboard is connected to the mouse port and/or a mouse is connected to the keyboard port. To ensure the user is aware that the IPPG has triggered the error state, the IPPG will utilize an error indicator light on the front of the switch. The error indicator light will be a red LED located near the keyboard and mouse ports of the IPPG (see Figure 18).

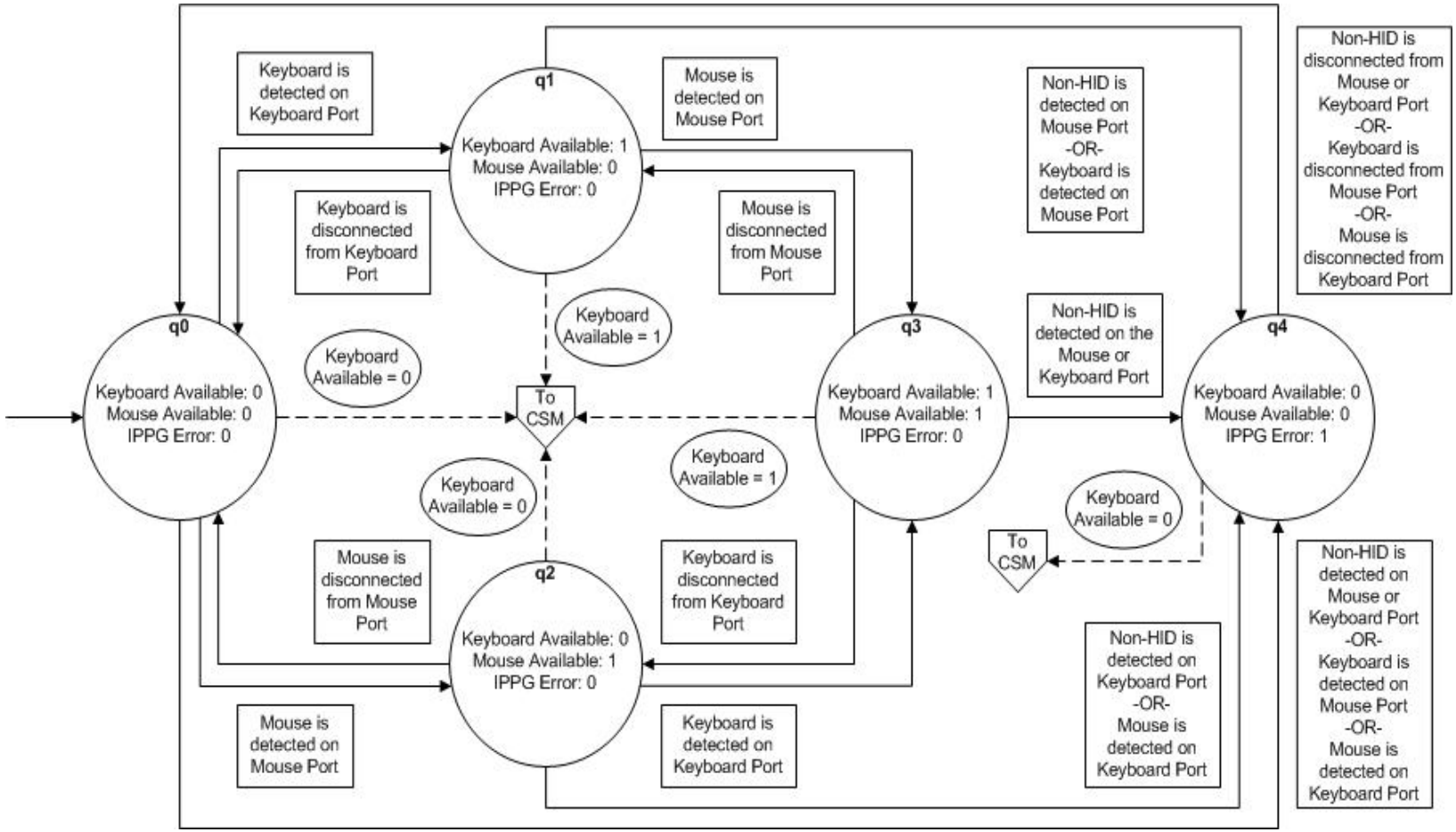

Figure 17. IPPG State Machine Including Error Handling State

Each state in Figure 16 and Figure 17 is defined by the variables in Table 4:

Variable Name	Value	Description
Keyboard Available	0	A keyboard is not connected to the keyboard port of the IPPG.
Keyboard Available	1	A keyboard is connected to the keyboard port of the IPPG.
Mouse Available	0	A mouse is not connected to the mouse port of the IPPG.
Mouse Available	1	A mouse is connected to the mouse port of the IPPG.
IPPG Error	0	The IPPG is not in the ERROR state. [LIGHT OFF]
IPPG Error	1	The IPPG has detected either a Non-HID connected to either the mouse port, keyboard port or both, a keyboard is connected to the mouse port, or a mouse is connected to the keyboard port. [LIGHT ON]

Table 4. IPPG's Basic Variables, Values, and Descriptions

Figure 18 displays the final design of the switch after all components have been defined with Table 5 describing the three different colored LEDs.

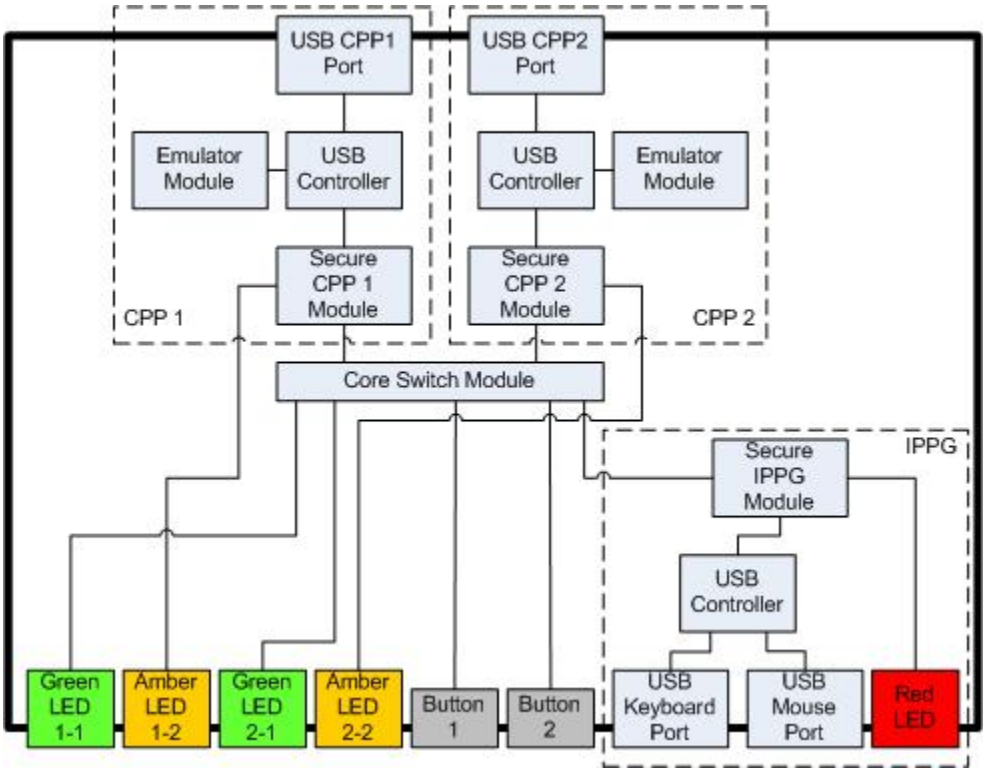

Figure 18. Switch Design–Final Iteration

Similar to the IPPG, the CPP is responsible for protecting the switch from USB packets sent from any of the attached computers that are not intended for a HID. The CPP will monitor the USB traffic coming from the attached computer and only allow USB Setup packets (Figure 5) or USB IN packets (Figure 3).

DC-19. The CPP will be coupled with a secure module to ensure only USB packets designated for an HID are transmitted through the switch.

To continue to ensure a positive user experience, the switch will need to provide visual indication for when a CPP has entered into the error state due to detected problems with any incoming USB packets. As each CPP is already in control of an Available Computer Indicator, the Available Computer Indicator will flash on and off while in the

error state. The error state can only be cleared by either disconnecting the computer causing the error condition or resetting the switch through a power cycle.

LED Color	Functional Description
Amber - On	A computer is attached to the CPP and powered on.
Amber - Off	A computer is not attached to the CPP or not powered on.
Amber - Flashing	The CPP has detected a USB packet destined for a non-HID and has entered the error state, disallowing further data from the computer attached to the CPP from flowing to the CSM.
Green - On	The CPP is selected and the path between the selected CPP and the IPPG is logically connected.
Green - Of	The CPP is not selected and no logical path exists between the CPP and the IPPG.
Green - Flashing	The CPP is selected, but the switch is still establishing the connection between the selected CPP and the IPPG.
Red - On	The IPPG is in the Error State.

Table 5. LED Colors and Functional Descriptions–Final

DC-20. The Computer Available Indicator will flash to denote an error condition detected by the CPP.

Figure 19 adds both an error condition to each CPP as well as a signal to the CSM if a computer is or is not connected to the CPP. When the error condition is detected, a new status for the Available Computer Indicator is also added, see Table 6.

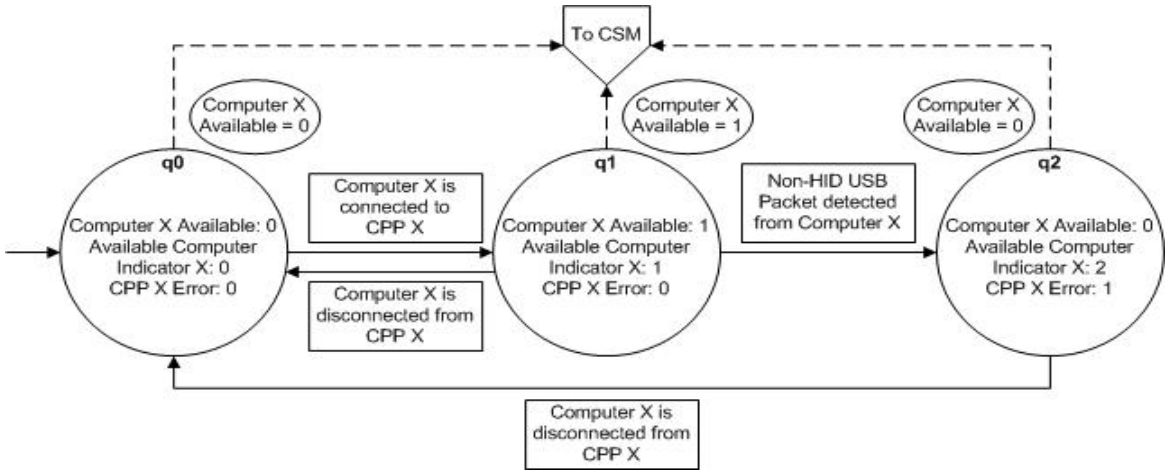

Figure 19. CPP State Machine Including Error Handling State

Table 6 defines the new values available to CPP X's variables with the addition of the error state:

Variable Name	Value	Description
Available Computer Indicator X	2	CPP X has detected an error caused by the connected Computer X.
CPP X Error	0	CPP X is not in the ERROR state.
CPP X Error	1	CPP X has detected an USB Packet originating from Computer X to be a non USB Setup or USB IN packet.

Table 6. CPP with Error State Variables, Values, and Descriptions

With the external interfaces properly secured, the switch will need to ensure the CSM only connects the currently selected computer with the attached keyboard and mouse. The CSM will be responsible for:

- Ensuring only the currently selected computer is allowed to communicate with the attached keyboard and mouse.
- Providing a secure transition state to ensure data is not inadvertently sent to the previously selected CPP.

To ensure the CSM is able to correctly initiate, maintain, and transition connections between any of the connected computers, the CSM will need to be aware of each CPP's state. Specifically, the CSM will need to know the status of "Computer X Available" from each CPP. If "Computer X Available" is set to 0, then the CSM shall ignore any user pressing Button X. However, if "Computer X Available" is set to 1, then the CSM will begin the process of allowing the attached Keyboard and/or Mouse to connect to the computer requested by the user.

DC-21. The CSM will consist of a secure module responsible for ensuring the correct CPP is allowed to communicate with the IPPG.

DC-22. The CSM will ignore any user pressing Button X if "Computer X Available" is set to 0.

Figure 20 visually represents the current state machine of the CSM.

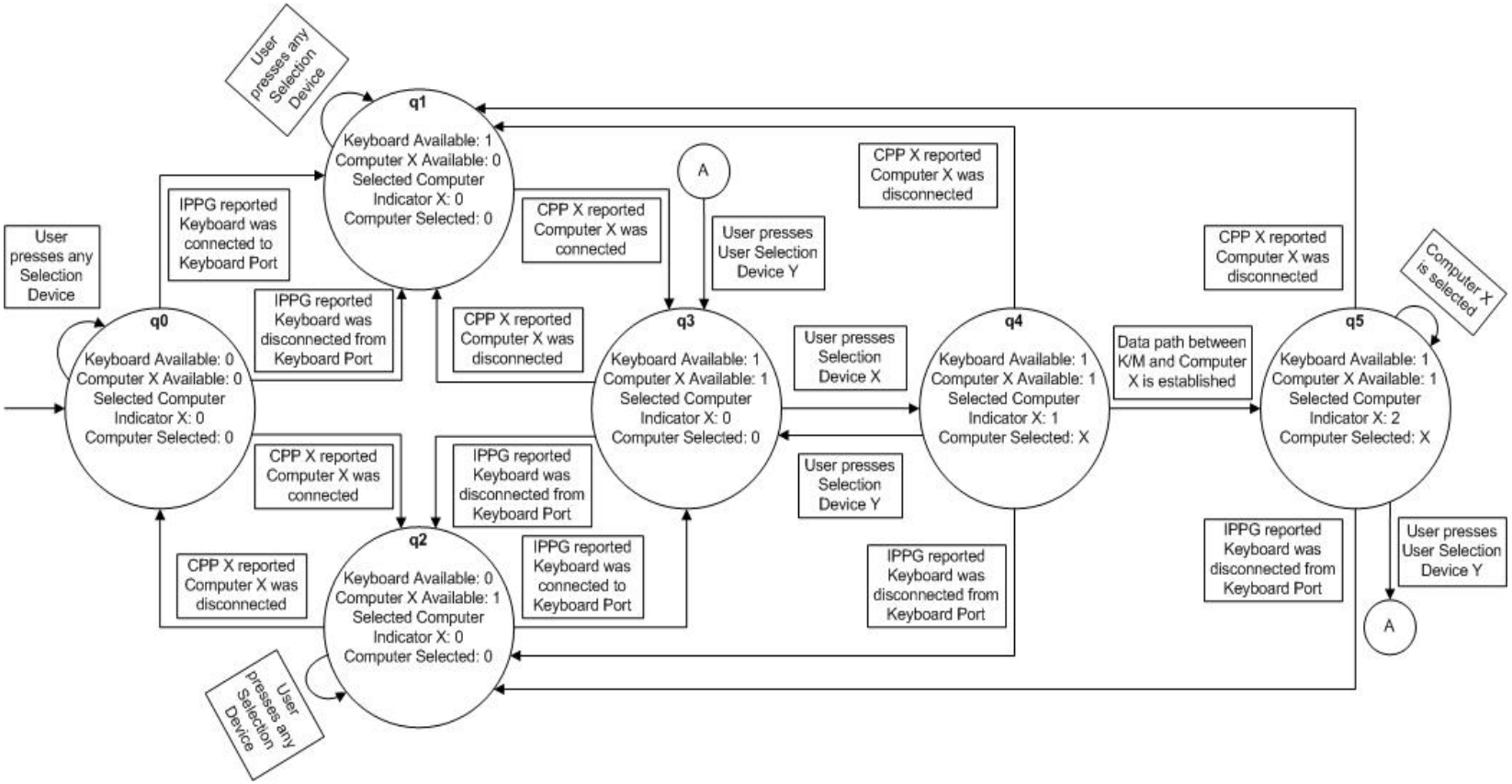

Figure 20. CSM Basic State Machine

Table 7 provides the CSM variables, values, and descriptions not previously defined:

Variable Name	Value	Description
Selected Computer Indicator X	0	The light is off. Computer X is not selected.
Selected Computer Indicator X	1	The light is blinking. Computer X is selected, but the data path between CPP X and the Keyboard and Mouse has not been established.
Selected Computer Indicator X	2	The light is on. Computer X can fully communicate with the attached Keyboard and Mouse.
Computer Selected	0	The user has not pressed the "User Selection Device X."
Computer Selected	X	The user has pressed the "User Selection Device X."
Keyboard Available	0	The IPPG has not yet reported a keyboard connected to the keyboard USB port or has reported the keyboard disconnected from keyboard USB port.
Keyboard Available	1	The IPPG has reported a keyboard connected to the keyboard USB port.

Table 7. CSM's Variables, Values, and Descriptions

The CSM will need to provide a secure transition state or set of states, Figure 21, which allows the IPPG and the attached keyboard and/or mouse to properly switch from interacting with one CPP to interacting with a different without allowing data intended for the previously selected CPP to reach the newly selected CPP.

DC-23. The CSM will provide a secure transition state or set of states allowing the IPPG to disconnect from the previously selected CPP and connect to the newly selected CPP.

The CSM will enter the transition state or states when the user presses one of the buttons on the front of the switch. The CSM will immediately cause the "Selected Computer Indicator" to begin blinking, providing a visual indicator to the user that the switch is in the process of connecting the desired CPP to the IPPG. If a different CPP was previously selected, the CSM will also immediately stop all communications between the previously selected CPP and the IPPG. In addition, the CSM will send the IPPG a "Flush" command informing the IPPG that it needs to clear the buffers contained in the attached USB Keyboard and Mouse. The CSM will then wait for the IPPG to respond that the command completed successfully or the timeout period has been exceeded. If the command times out, the CSM will turn off the "Selected Computer Indicator X," signaling that the user will need to reselect the CPP or select a different CPP. If, however, the command succeeds, the CSM will allow the regular USB SETUP process to occur between the selected CPP and the keyboard/mouse, "q6" in Figure 21. Once the Setup process has completed, the CSM will continuously illuminate the "Selected Computer Indicator X" until the switch is powered off, the computer connected to CPP X is disconnected, or the user selects a different CPP with which to interact.

DC-24. The CSM will send a "Flush" command to the IPPG as part of the secure transition state or set of states.

DC-25. The CSM will wait for the IPPG to confirm completion of the "Flush" command before allowing the newly selected CPP to be connected to the IPPG or time out reverting back to the "none selected" state.

Figure 21 depicts the state machine of the CSM to include the transition states.

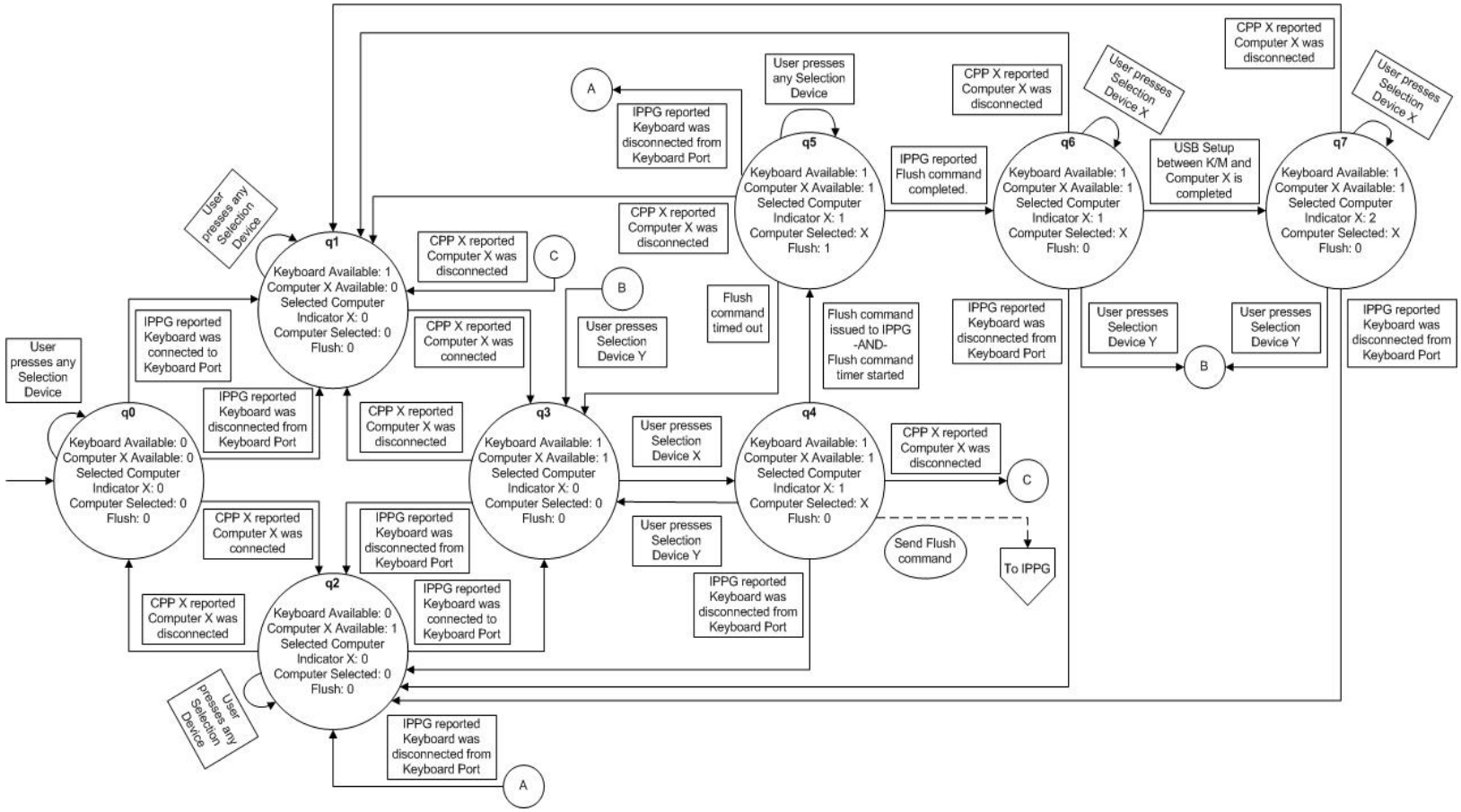

Figure 21. CSM State Machine Including Transition States

Table 8 provides the updated CSM variables, values, and descriptions not previously defined:

Variable Name	Value	Description
Flush	0	The "Flush" command has not been issued, has completed, or has timed out.
Flush	1	The "Flush" command is currently running.

Table 8. CSM's Updated Variables, Values, and Descriptions

As part of the transition states, the IPPG, Figure 22, is responsible for ensuring the buffers of the attached keyboard and mouse are cleared before a new connection can be established with a CPP and reporting back to the CSM once the process is complete. The IPPG will use the same "Flush" command whenever a keyboard or mouse is plugged-in and detected by the IPPG to ensure a clean start state for the device's buffers before alerting the CSM that a keyboard is available. The IPPG will use the same "Keyboard Available" signal to alert the CSM when the "Flush" command has completed.

DC-26. The IPPG will ensure the buffers of the attached keyboard and mouse are properly flushed insuring residual data is removed.

DC-27. The IPPG will report back to the CSM once the "Flush" command has completed successfully.

Figure 22 depicts the state machine of the IPPG with the Flush states.

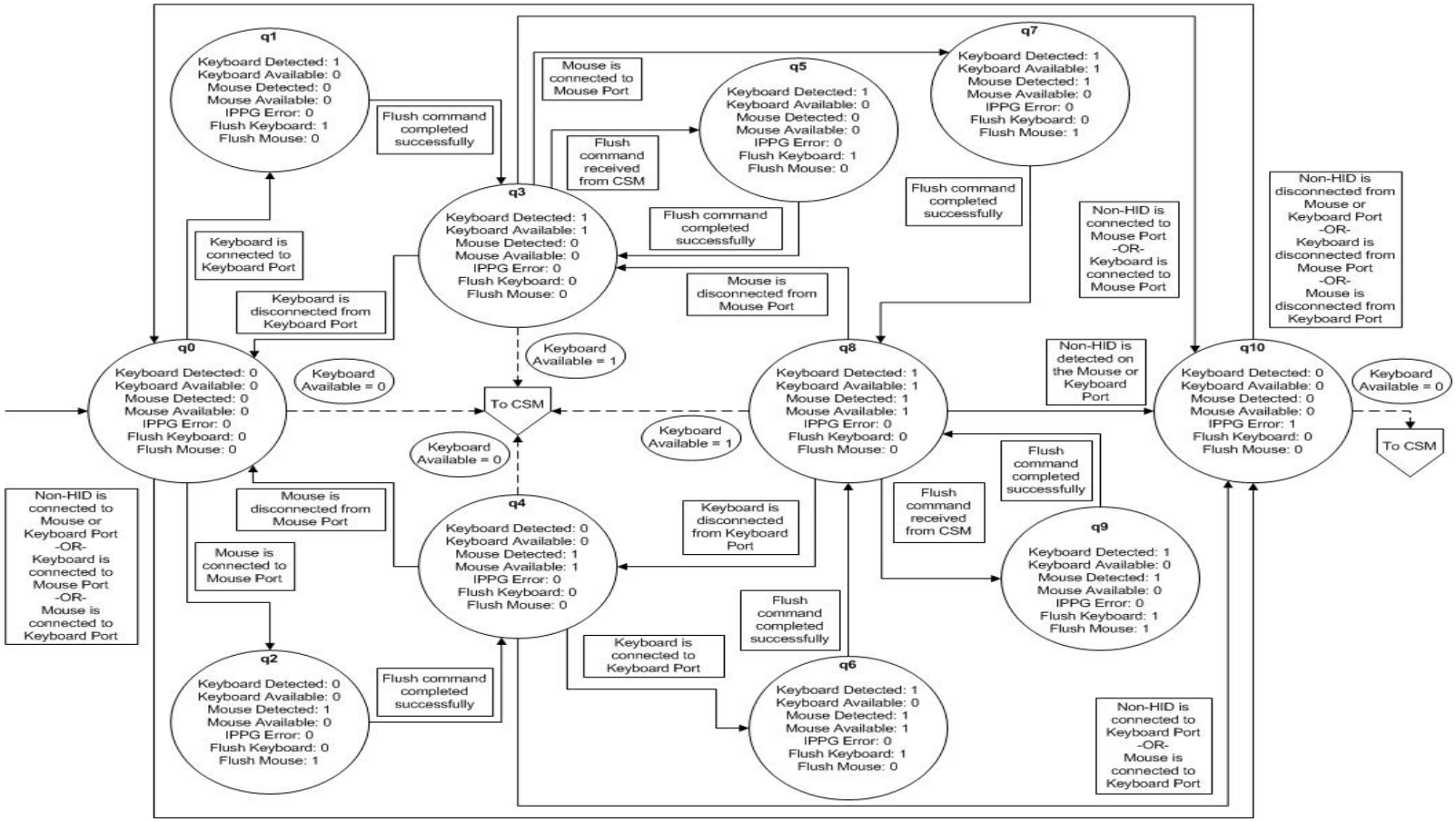

Figure 22. IPPG State Machine Including Flush Command States

Table 9 provides the updated IPPG's variables, values, and descriptions not previously defined:

Variable Name	Value	Description
Flush Keyboard	0	The "Flush" command has not been issued or has completed.
Flush Keyboard	1	The "Flush" command is currently running.
Flush Mouse	0	The "Flush" command has not been issued or has completed.
Flush Mouse	1	The "Flush" command is currently running.

Table 9. IPPG's Updated Variables, Values, and Descriptions

The following diagrams (Figure 23 and Figure 24) represent the state diagram of the CSM of a 2-port keyboard/mouse switch. A 2-port switch would allow at most 2 computers to share a single keyboard and mouse. These diagrams were split and references were used to aid insertion into this document as well as prevent transition lines from overlapping.

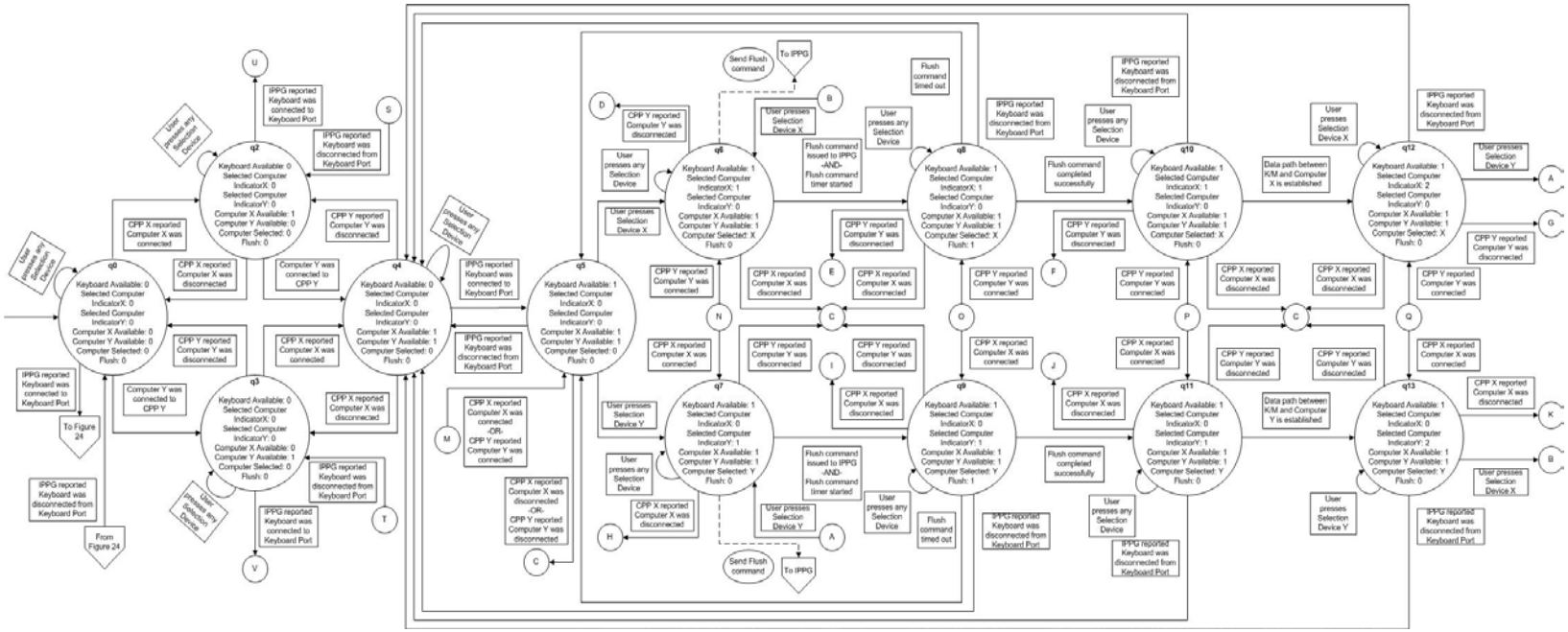

Figure 23. (A) CSM State Machine–2 CPPs

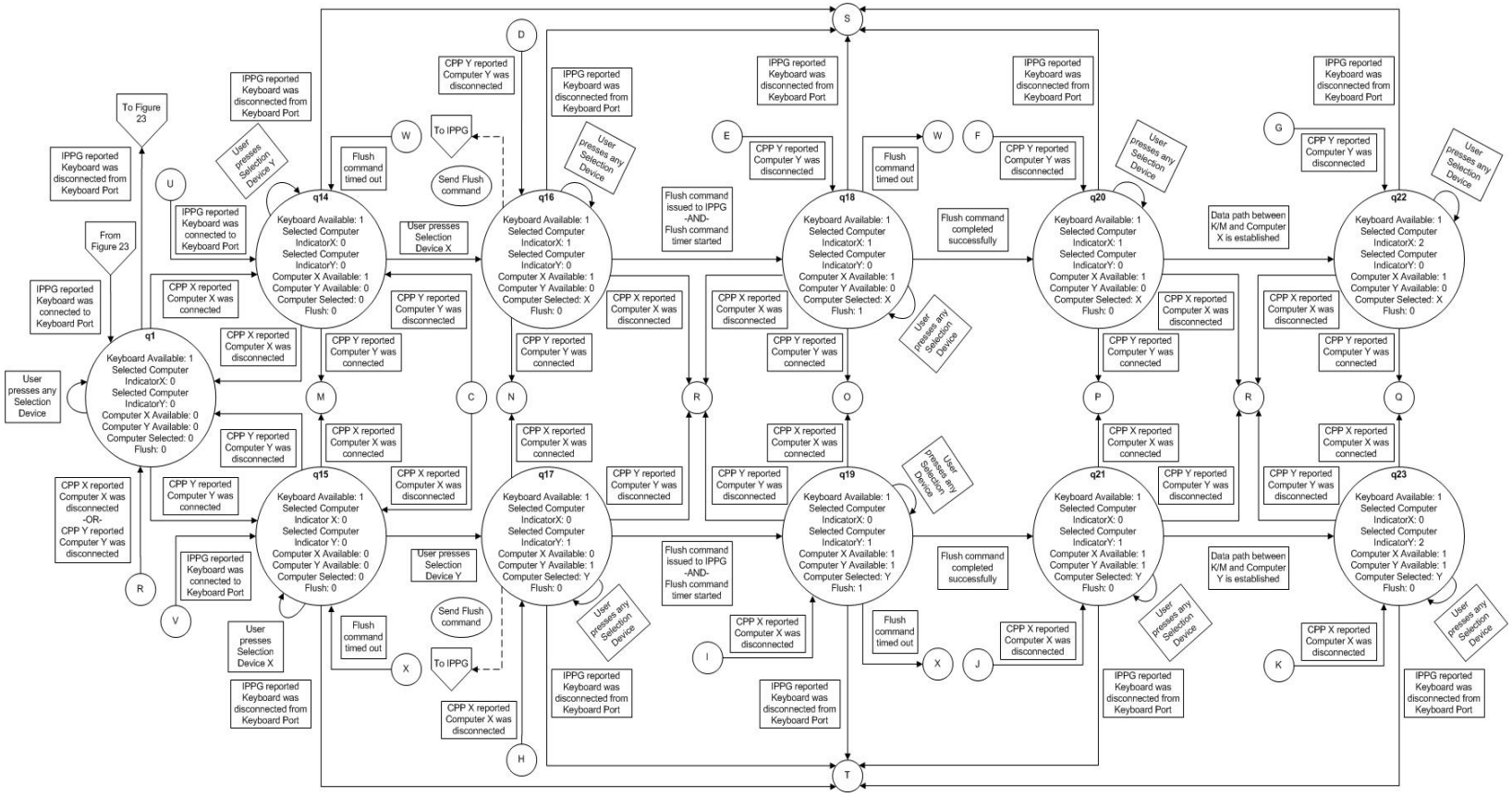

Figure 24. (B) CSM State Machine–2 CPPs

VII. ANALYSIS, CONCLUSION, AND FUTURE RESEARCH

A. ANALYSIS

In this paper, our goal was to define the requirements of a highly robust peripheral sharing switch for HIDs and to show, at a high level, a secure design. Each requirement was examined for security implications and derived requirements or design choices were made to adequately cover potential security flaws.

The requirements and the data flow diagrams included in Appendix B were instrumental in defining the state diagrams for the main components (IPPG, CPP, and CSM) of the switch. The three state diagrams Figure 19, Figure 22, and Figure 23/Figure 24 combine to create the complete state diagram for the highly robust peripheral sharing switch. Each component evolved as derived requirements and design choices combined to describe the data flows between the IPPG and the CSM, the CSM and each CPP, and the IPPG and each CPP. Once internal and external data flows for each component were clearly defined, the state diagrams were developed to clearly show the state of each component at any given time.

The state diagram for the CSM (Figure 23 and Figure 24), with the help of the state transition table (Table 10), clearly show that at no time can the switch enter into a state where more than one CPP is communicating with the IPPG at the same time and a flush command is always issued before a connection is allowed between a CPP and the IPPG.

The first two requirements are shown to be true in the CSM state machine depicted in Figure 23 and Figure 24. That is, no state is capable of allowing the "Computer Selected" to be both X and Y, and the states q12, q13, q22, and q23 are only accessible if the CSM first transitions through q6, q7, q16, or q17. The third requirement will be shown to hold true with the combination of the CSM state machine and the implementation of the CSM.

Table 10 is the state transition table for the CSM when two computers (X and Y) are connected to the switch. The first column contains the states and the first row contains the events associated with CSM. The intersection of each row and column lists the next state, if the event is valid for the current state, and any associated actions that will occur upon transitioning to the next state. Table 11 contains the legend that maps the actions in Table 10 to their descriptions.

	Events											
State	User Presses Selection Device X	User Presses Selection Device Y	IPPG Reports Keyboard is Available	IPPG Reports Keyboard is Not Available	CPP X Reports Computer X is Available	CPP X Reports Computer X is Not Available	CPP Y Reports Computer Y is Available	CPP Y Reports Computer Y is Not Available	Flush Command Sent to IPPG	Flush Command Times Out	Flush Command Successful	USB Setup Process Completed
q0	q0	q0	q1	q0	q2	q0	q2	q0	-	-	-	-
q1	q1	q1	-	q0	q14	q1	q15	q1	-	-	-	-
q2	q2	q2	q14	q2	-	q0	q4	q2	-	-	-	-
q3	q3	q3	q15	q3	q4	q3	-	q0	-	-	-	-
q4	q4	q4	q5	q4	-	q3	-	q2	-	-	-	-
q5	q6/A	q7/B	-	q4	-	q15	-	q14	-	-	-	-
q6	q6	q6	-	q4/C	-	q15/C	-	q16	q8/E	-	-	-
q7	q7	q7	-	q4/D	-	q17	-	q14/D	q9/E	-	-	-
q8	q8	q8	-	q4/C	-	q15/C	-	q18	-	q5/C	q10/F	-
q9	q9	q9	-	q4/D	-	q19	-	q14/D	-	q5/D	q11/F	-
q10	q10	q10	-	q4/C	-	q15/C	-	q20	-	-	-	q12/G
q11	q11	q11	-	q4/D	-	q21	-	q14/D	-	-	-	q13/H
q12	q12	q7/C,B	-	q4/I	-	q15/C	-	q22	-	-	-	-
q13	q6/D,A	q13	-	q4/J	-	q23	-	q14/D	-	-	-	-
q14	q16/A	q14	-	q2	-	q1	q4	-	-	-	-	-
q15	q15	q17/B	-	q3	q4	-	-	q1	-	-	-	-
q16	q16	q16	-	q2/C	-	q1/C	q6	-	q18/E	-	-	-

State	Events											
	User Presses Selection Device X	User Presses Selection Device Y	IPPG Reports Keyboard is Available	IPPG Reports Keyboard is Not Available	CPP X Reports Computer X is Available	CPP X Reports Computer X is Not Available	CPP Y Reports Computer Y is Available	CPP Y Reports Computer Y is Not Available	Flush Command Sent to IPPG	Flush Command Times Out	Flush Command Successful	USB Setup Process Completed
q17	q17	q17	-	q3/D	q7	-	-	q1/D	q19/E	-	-	-
q18	q18	q18	-	q2/C	-	q1/C	q8	-	-	q14/C	q20/F	-
q19	q19	q19	-	q3/D	q9	-	-	q1/D	-	q15/D	q21/F	-
q20	q20	q20	-	q2/C	-	q1/C	q10	-	-	-	-	q22/G
q21	q21	q21	-	q3/D	q11	-	-	q1/D	-	-	-	q23/H
q22	q22	q22	-	q2/C	-	q1/C	q12	-	-	-	-	-
q23	q23	q23	-	q3/D	q13	-	-	q1/D	-	-	-	-

Table 10. CSM State Transition Table

Actions	
A	User Selection Indicator X begins Flashing.
B	User Selection Indicator Y begins Flashing.
C	User Selection Indicator X is Off.
D	User Selection Indicator Y is Off.
E	Flush Command timer started.
F	The USB Setup Process is allowed to commence.
G	User Selection Indicator X is On.
H	User Selection Indicator Y is On.

Table 11. CSM State Transition Table Actions

Status	Functional Description
On	The User has pressed the User Selection Device for the CPP and the path between the selected CPP and the IPPG is logically connected.
Off	The User has not pressed the User Selection Device for the CPP or has selected a different CPP. Therefore, no logical path exists between the CPP and the IPPG.
Flashing	The User has pressed the User Selection Device designating the CPP is selected, but the switch is still establishing the connection between the selected CPP and the IPPG.

Table 12. User Selection Indicator Functional Descriptions–Final

B. CONCLUSION

The design process of a secure keyboard/mouse Switch led to the establishment of the following fundamental requirements:

1. Only a single computer connected to the switch is allowed to communicate with the attached keyboard and mouse at any given point in time.
2. A “Flush” command must precede any connection between an attached computer and the keyboard and mouse.
3. No two CPPs are allowed to communicate with each other.

An inference can also be made that by properly implementing the requirements defined above the implementer will ensure that no data from any CPP will be allowed to pass through to any other CPP.

The idea of a secure keyboard/mouse switch, at first blush, appears to be simple and straightforward system. However, the complex set of states as shown in Figure 23 and Figure 24 that needed to be developed and understood to achieve the security goals show that the system is neither simple nor straightforward.

In the course of developing these requirements, certain assumptions had to be made to insure the security of the system. These assumptions were:

1. The attached keyboard will comply with the "Flush" command.
2. The switch will be able to identify the peripheral USB devices attached to the switch as a keyboard and mouse.
3. The switch will be able to identify all communications between the peripheral devices and the computers as legitimate USB traffic intended for or from a HID.

C. FUTURE RESEARCH

The process of designing and building a secure keyboard and mouse switch begins with the definition of requirements and mapping those requirements to rudimentary state machines. The independent review provided here establishes some of the assurances, but the next step is mapping the assurances to an implementation of the design. The state machines should also go through a more formal analysis to ensure the appropriate level of assurance required by the user. This analysis can be performed by tools such as Harel Statecharts[7] or Alloy[8] for semiformal analysis or for a formal analysis, ACL2.[9] The hardware used in the implementation of the design should also be tested for high robustness.

---

7 Harel, David. "Statecharts: A visual formalism for complex systems." *Science of Computer Programming* 8 (1987): 231-274. PDF.

8 Jackson, Daniel. "Alloy: a lightweight object modeling notation." *ACM Transaction on Software Engineering and Methodology* 11.2 (2002). PDF.

9 Kaufmann, Matt, and Moore, J. Strother. *"Industrial Proofs with ACL2."* PDF File.

To be a true KVM switch, the switch should also be able to accurately display any of the connected computers on a single video display device. The uniqueness of the video signal may introduce further requirements that will need to be added to the already defined requirements in this paper.

Along with the video signal, there are other technologies currently in use by commercially available KVM switches that may be of interest to further the advancement of a secure KVM. These technologies include KVM over IP and Multiuser KVM switches. KVM over IP switches essentially increase the distance that a user can be from the physical computer. These switches involve two separate hardware devices that connect via a standard network cable (i.e., CAT5). The main switch device plugs directly into the computers via USB cables and video cables, and when compared to the design described in this paper would include the CPPs and the CSM. The secondary device would connect to the keyboard, mouse, and video display device and provide a means for the user to select from which computer to send and receive date. The secondary device would also provide a means to display which computer is selected, and may or may not include indicators to show which computers are selectable. In reference to our design, the secondary device would include the IPPG, buttons, and any indicator lights. Multiuser KVM switches are generally KVM over IP switches that have the added functionality to allow more than one console or user access to the attached computers through the switch.

APPENDIX A: COMBINED REQUIREMENTS

C-1. **The switch shall enable communication between a keyboard and a mouse and at most one computer connected to the switch at a given point in time.**

C-2. **The switch shall not allow communications with peripheral devices connected to the IPPG other than a keyboard on the keyboard port and a mouse on the mouse port.**

C-3. **The switch shall allow the user to select the computer to be paired with the attached keyboard and mouse.**

C-4. **The switch shall, indicate which computer, if any, is paired with the attached keyboard and mouse.**

C-5. **The computers connected to the switch shall be able to boot normally.**

D-1. **The switch shall be powered.**

DC-1. **Standard wall power will be used to power the device.**

DC-2. **Conventional COTS IT and electronics shall be utilized.**

D-2. **Core requirements C1, C-3, C-4, and C-5 are applicable only when the switch is powered on.**

D-3. **The switch shall not allow communication between the keyboard and a mouse and any computer when the switch is powered off.**

D-4. **As viewed externally, the switch shall have a finite set of atomic state transitions.**

D-5. **The switch shall visually present a finite set of internal states of the switch.**

DC-3. **The switch will support at least two computers.**

DC-4. **The switch will present the internal states identifying which computer is paired through the switch.**

DC-5. The main components of the switch will consist of an IPPG, a finite number of CPPs, a switch module, and a user interface (see Figure 12).

D-6. The switch shall only pair with an active computer connected to a CPP.

DC-6. The switch will indicate which computers are available.

D-7. At no point in time shall two CPPs be allowed to communicate through the core switching module.

D-8. The switch shall protect the integrity of the communications between computer selection devices, the selected computer indicators, the CPPs, and the CSM itself.

DC-7. Protective Design.

DC-8. The external ports to the switch will be defined by the USB standard.

DC-9. Each CPP and the IPPG will be coupled with a USB Controller.

DC-10. The switch will utilize physical buttons for computer selection.

DC-11. The switch will utilize a GREEN LED light to visually represent the currently selected CPP.

DC-12. The switch will utilize an AMBER LED light to visually represent an active CPP.

DC-13. The GREEN LED will blink while the path between a CPP and the IPPG is setup.

DC-14. The buttons on the front panel will only communicate with the CSM

DC-15. The green LED lights will be controlled by the same secure processor as the buttons.

DC-16. Each CPP will be coupled with an EM designed to emulate USB keyboard functionality to ensure any attached computer can properly boot.

DC-17. Each EM will be powered via USB from the attached computer while the switch is powered off.

DC-18. The IPPG will be coupled with a secure module to ensure only USB packets originating from a HID are transmitted through the switch.

DC-19. The CPP will be coupled with a secure module to ensure only USB packets designated for a HID are transmitted through the switch.

DC-20. The Computer Available Indicator X will flash to denote an error condition detected by the CPP X.

DC-21. The CSM will consist of a secure module responsible for ensuring the correct CPP is allowed to communicate with the IPPG.

DC-22. The CSM will ignore any user pressing Button X if "Computer X Available" is set to 0.

DC-23. The CSM will provide a secure transition state or set of states allowing the IPPG to disconnect from the previously selected CPP and connect to the newly selected CPP.

DC-24. The CSM will send a "Flush" command to the IPPG as part of the secure transition state or set of states.

DC-25. The CSM will wait for the IPPG to affirm completion of the "Flush" command before allowing the newly selected CPP to be connected to the IPPG or time out reverting back to the "none selected" state.

DC-26. The IPPG will ensure the buffers of the attached keyboard and mouse are properly flushed to insure residual data is removed.

DC-27. The IPPG will report back to the CSM once the "Flush" command has completed successfully.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: DATA FLOW DIAGRAMS

Figure 25 depicts the potential data flows while the switch is powered on with a keyboard connected to the keyboard port, a mouse connected to the mouse port, and computer X is powered on and connected to CPP X.

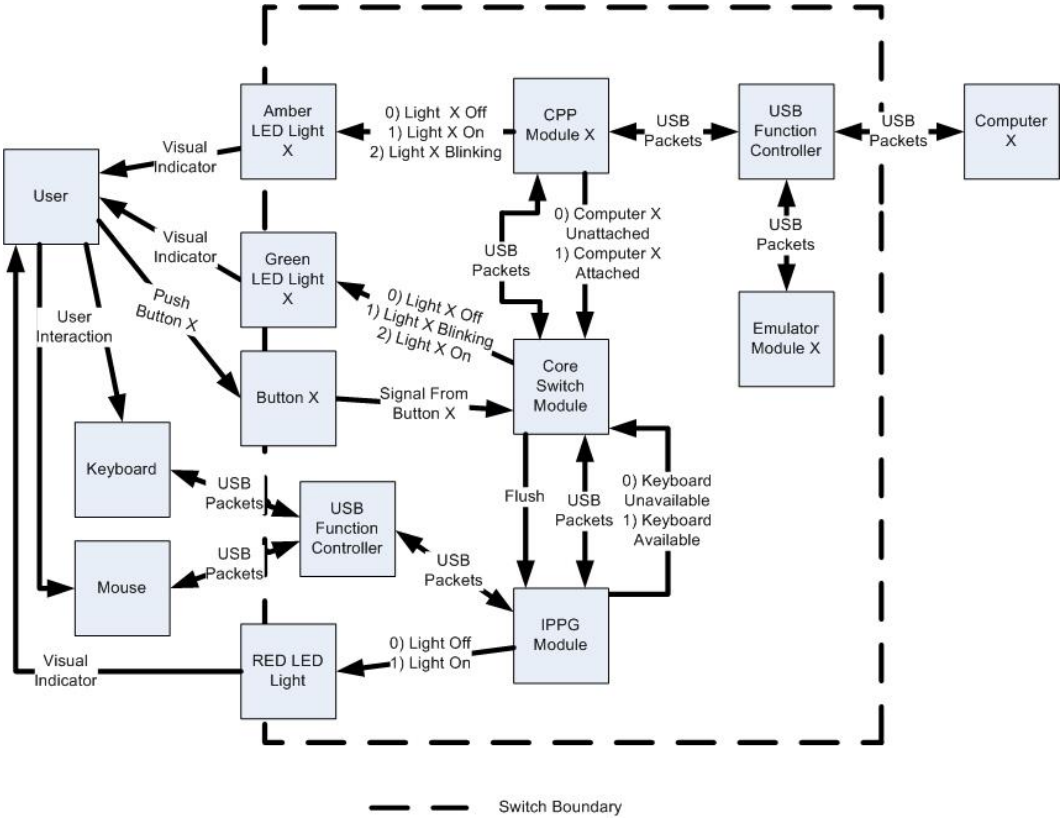

Figure 25. Potential Data Flows with Switch Powered On

Figure 26 shows the data flows available through the switch while the switch is powered off. Computer X will provide power over USB to the Emulator Module, which in turn will emulate a generic keyboard allowing Computer X to boot properly.

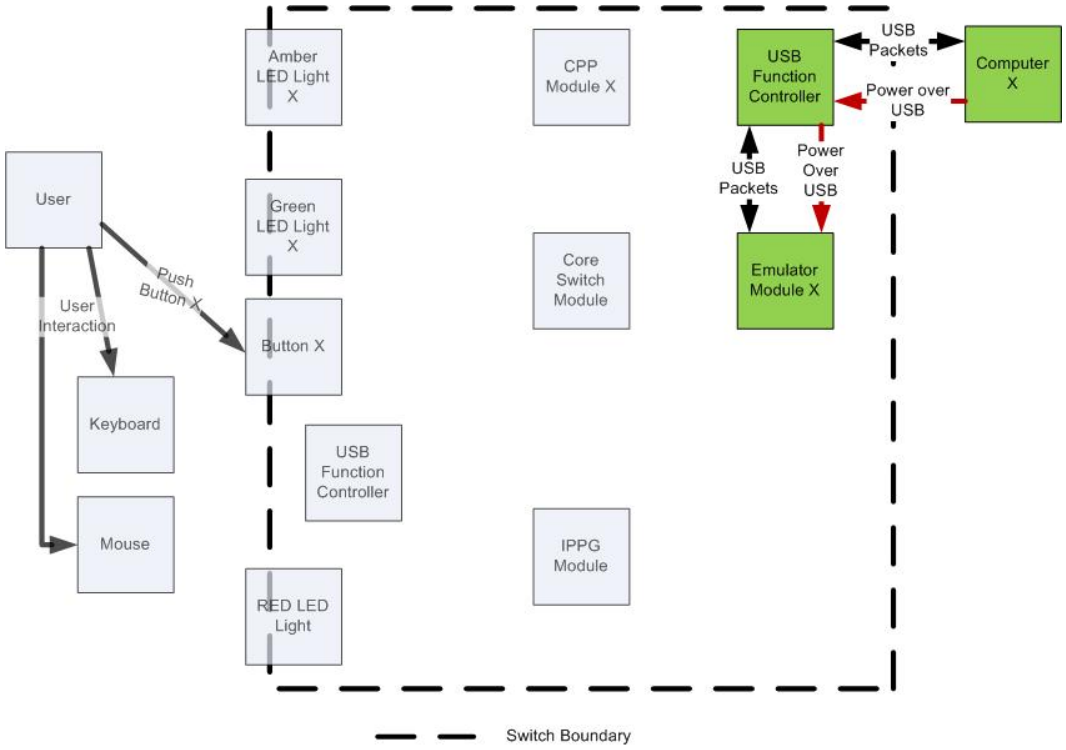

Figure 26. Potential Data Flows with Switch Powered Off

Figure 27 shows the data flows available through the switch when the IPPG enters the error state. The IPPG will deny any further USB packets from flowing from the attached peripheral devices into the CSM, thus preventing the USB packets from potentially reaching the currently selected computer. While in the error state the IPPG will turn on the Red LED.

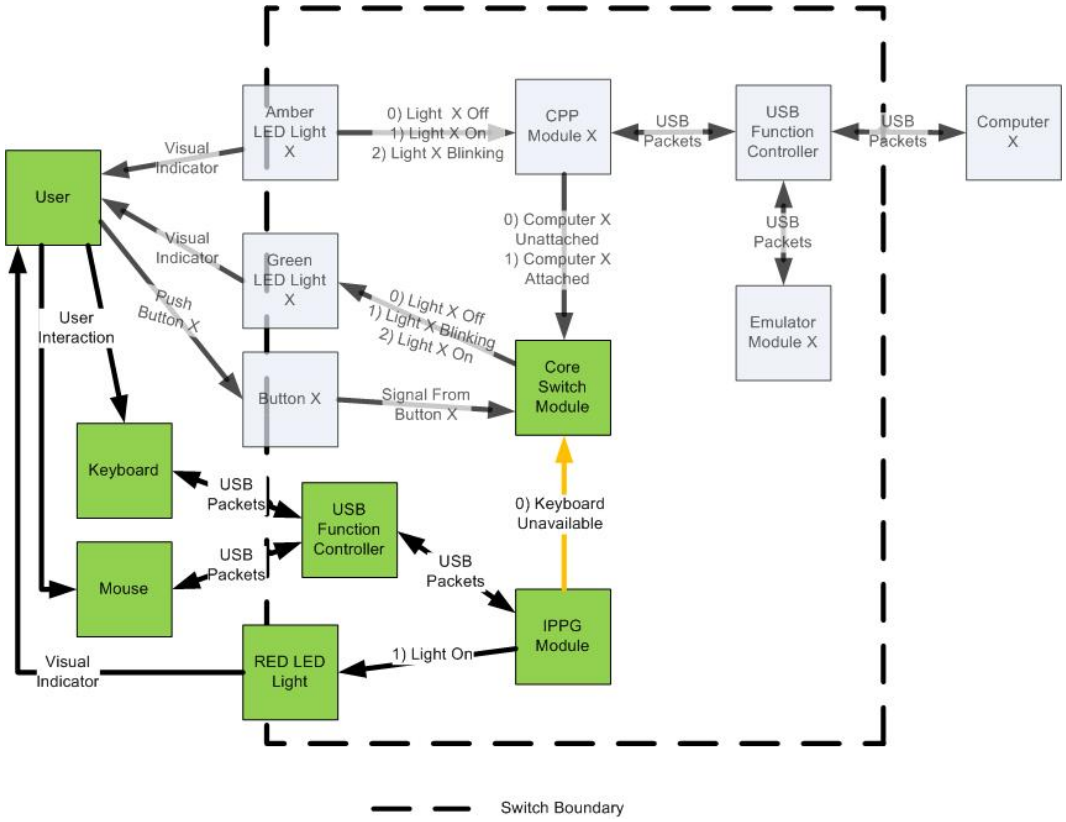

Figure 27. Potential Data Flows–IPPG Module Error

Figure 28 shows the data flows available through the switch when a CPP enters the error state. The CPP will report that the computer is no longer available to the CSM and deny any further USB packets from flowing from the attached computer into the CSM, thus preventing the USB packets from potentially reaching the connected peripheral devices. While in the error state the CPP will cause the Amber LED to blink.

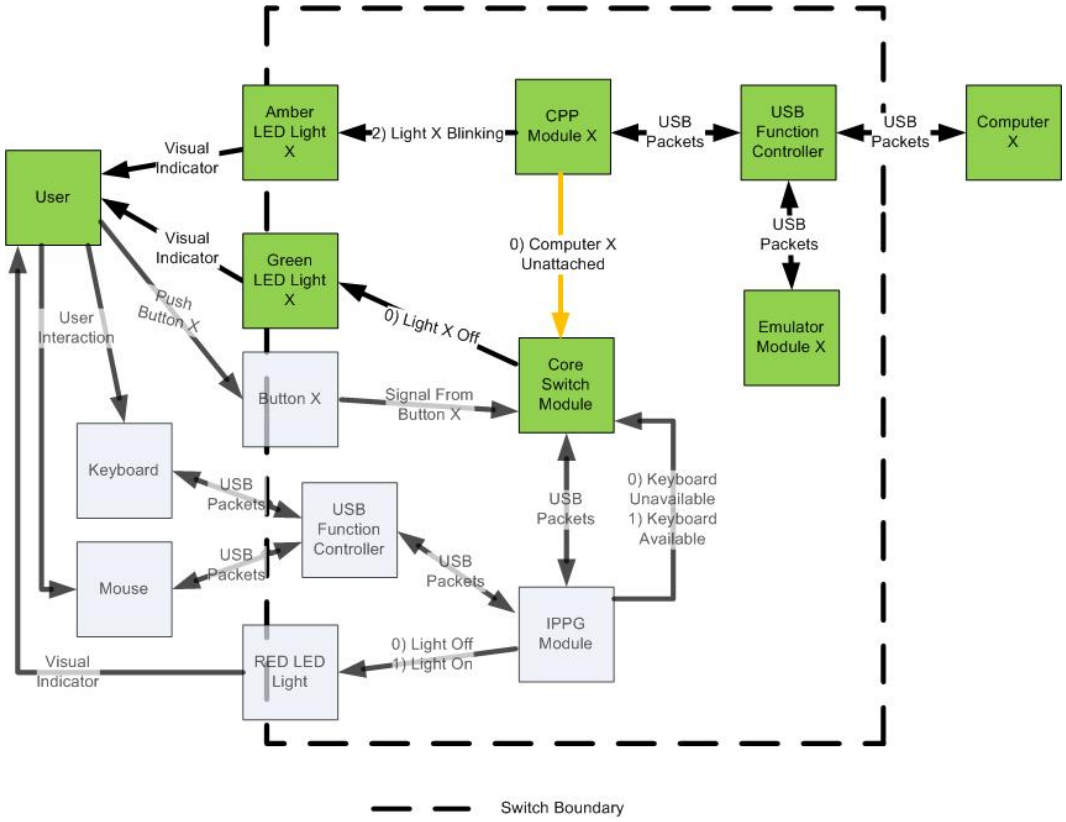

Figure 28. Potential Data Flow–CPP X Module Error

Figure 29 represents the potential data flows available while the switch is powered on, there is potentially an attached keyboard and mouse, and one or more computers are attached and powered on but none are selected.

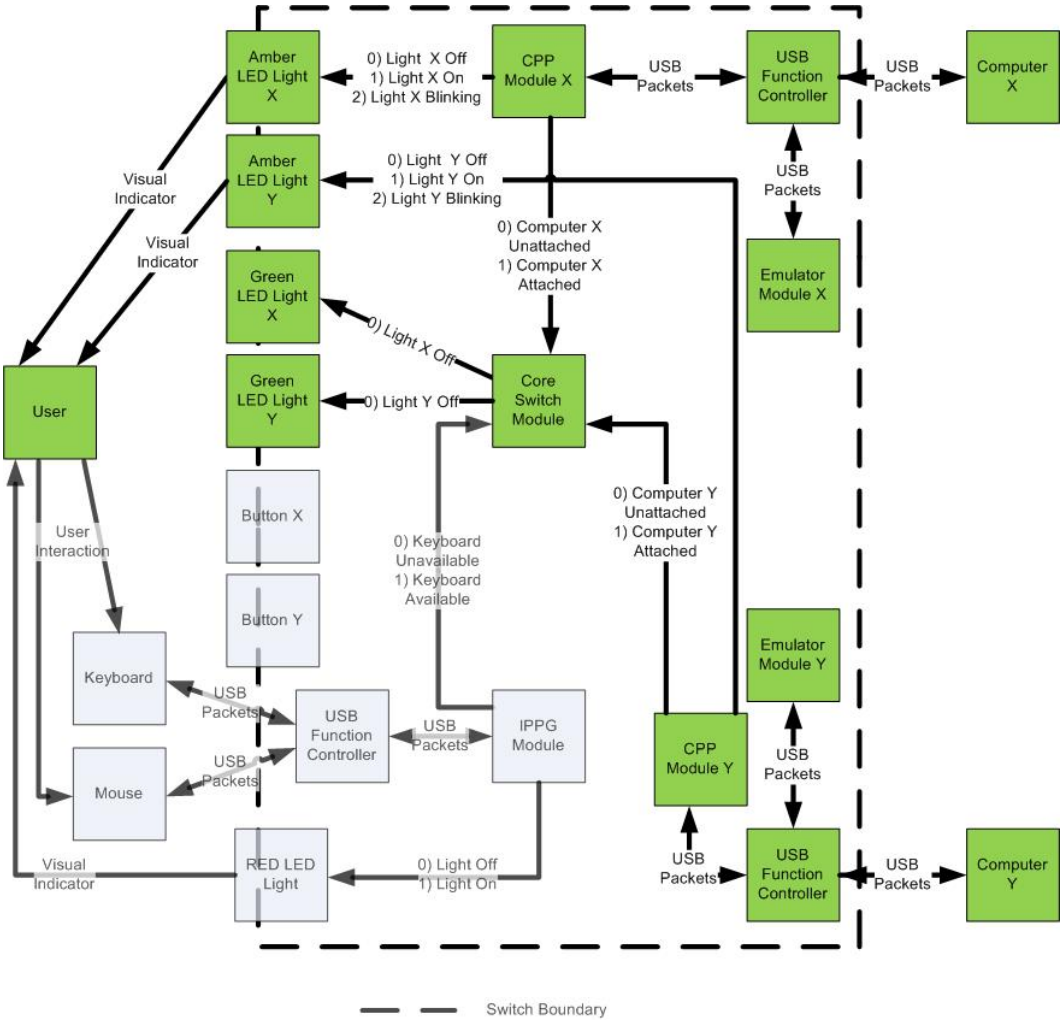

Figure 29. Potential Data Flows–No CPP Selected

Figure 30 represents the potential data flows available while the switch is powered on, there is an attached keyboard and mouse, and a computer is attached, powered on, and selected.

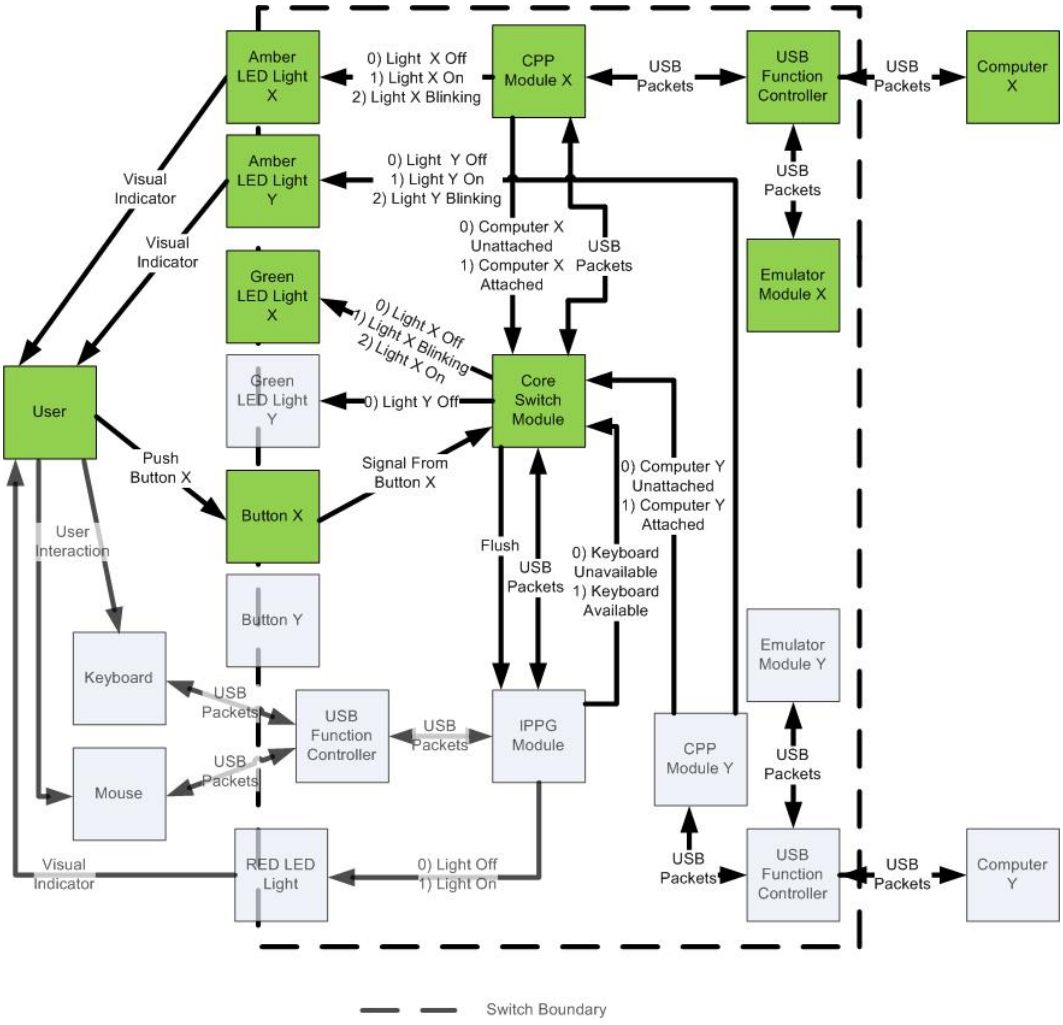

Figure 30. Potential Data Flows–Button X Selected

LIST OF REFERENCES

[1] D. L. Staneszewski, LtCol. "One Box – 1 Wire (OB1) United States Central Command" [PowerPoint slides]. United States Central Command, MacDill Air Force Base, FL, 06 May 2008.

[2] Information Assurance Directorate. "Peripheral Sharing Switch (PSS) for Human Interface Devices Protection Profile," 07 September 2010. PDF File.

[3] B. Nuseibeh. "Weaving Together Requirements and Architecture." *IEEE Computer*, vol. 34, no. 3, pp. 115–117, March 2001. PDF File.

[4] R. Martin. "Iterative and Incremental Development." *Engineering Notebook Column*. C++ Report, February 1999. PDF File.

[5] W. Scacchi. "Process Models in Software Engineering." October 2001, Institute for Software Research. University of California, Irvine. PDF File.

[6] National Information Assurance Partnership. "Part 3: Security assurance components." *Common Criteria for Information Technology Security Evaluation.* July 2009. PDF File.

[7] National Security Agency. "Analysis of the Avocent SC4-UAD KVM Switch," April 2007. PDF File.

[8] National Information Assurance Partnership. "Common Criteria Evaluation and Validation Scheme Validation Report Tenix Defence Pty Ltd Interactive Link Version 5.1." Version 1.0. 19 August 2005. PDF File.

[9] "Belkin® OmniView™ Secure KVM Models: F1DN102U F1DN104U F1DN108U Security Target EAL4 augmented ALC_FLR.3." Version 1.1. 17 February 2009. PDF File.

[10] "Validation Report Belkin® OmniView™ Secure KVM Switch (F1DN102U, F1DN104U, F1DN108U)." InfoGard Laboratories, 25 February 2009. PDF File.

[11] *Universal Serial Bus Specification Revision 2.0.* USB Implementers Forum. April 2000. PDF File.Who owns this?

[12] "Formal description techniques (FDT) – Message Sequence Charts." *Series Z: Languages and General Software Aspects for Telecommunications Systems* Z.120, February 2011. PDF File.

[13] Department of Defense Systems Management College. "Systems Engineering Fundamentals," January 2001. PDF File.

[14] Department of Defense. "Department of Defense Trusted Computer System Evaluation Criteria." December 1985. PDF File.

[15] National Information Assurance Partnership. "Part 1: Introduction and general model." *Common Criteria for Information Technology Security Evaluation*. July 2009. PDF File.

[16] National Information Assurance Partnership. "Part 2: Functional security components." In *Common Criteria for Information Technology Security Evaluation*, July 2009. PDF File.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Douglas Tanner
   SPAWAR Systems Center Atlantic
   North Charleston, SC

4. Jennifer Guild
   SPAWAR Systems Center Atlantic
   North Charleston, SC

5. Mark Vanfleet
   National Security Agency
   Fort Meade, MD

6. John Loucades
   National Security Agency
   Fort Meade, MD

7. Matt Benke
   National Security Agency
   Fort Meade, MD

8. Loren Peitso
   Naval Postgraduate School
   Monterey, CA

9. Bret Michael
   Naval Postgraduate School
   Monterey, CA