

# Model Railroad Computer Control

(How I am going to write my Train Program)

**Matt Katzer  
Portland, Or.**



# Agenda

- **Philosophy**
- **Hardware requirements**
- **Command control software available**
- **How should I write software**
- **Engine-Commander™ example**
  - Marklin (AC/DC) example
  - NMRA DCC example
- **Shareware software**
  - Compuserve
  - The Commander<sub>sm</sub> BBS
- **Demo**



# Why are you here

- Clinic will focus on writing programs to control your railroad....
  - we will talk about PC's
  - programming languages
  - example programs
- What are your expectations?

# Philosophy

- **Computer Controlled**
  - The computer controls the routes of the trains
  - the operator runs his/her layout from the computer
- **Computer Monitored**
  - the computer is a tool of the modeler
  - the computer is used to manage events
  - the computer does not control!



**I like to write software, but I want to run trains and use computers monitor the layout and to enhance the fun**

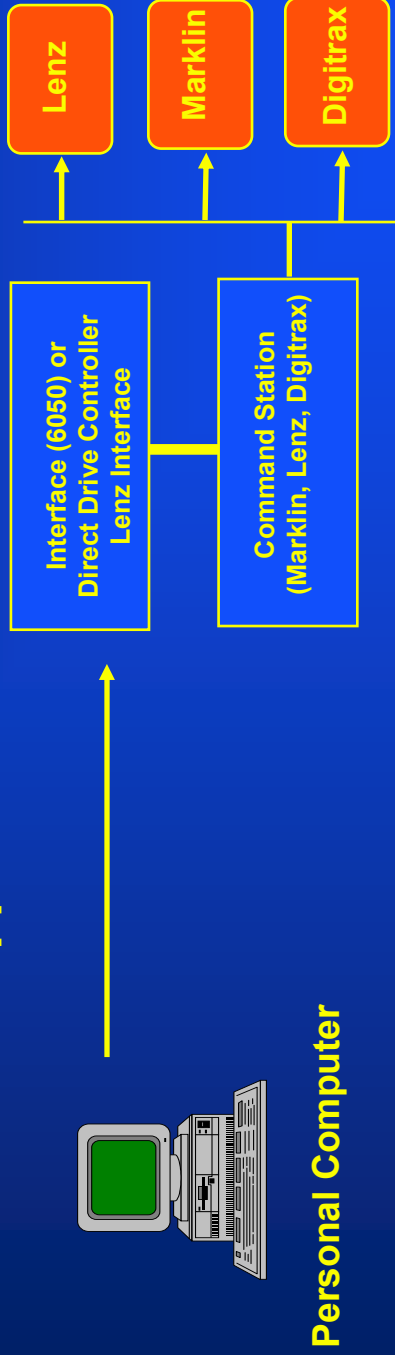


# Hardware Requirements

- **Apple versus IBM**
  - a religious battle
  - IBM has 70% PC market versus apples 21%
  - Almost all new software supports PC's
- **What type of PC hardware should you buy?**
  - Intel i486dx2-66 or higher processor
  - 8 Mbytes of system memory (16 Mbytes preferred)
  - 380 - 540 Mbyte (SCSI) hard disk
  - 3.5" Floppy drive
  - CDROM Drive (SCSI)
  - Sound Blaster Pro (SCSI version)
  - VGA graphics card (with 800 x 600 support)
  - VGA color monitor with SVGA support
- **What you should not buy**
  - 286 or 386 PC's
  - systems that contain less than 8 Mbytes

# Railroad Requirements

- **Must have NMRA DCC compatible engines**
  - either a Marklin C82, Digitrax, Lenz or Digital Plus system
- **Two type of interfaces supported**
  - PC/mac compatible serial interface (9.6K)
  - Direct drive support via command station



Personal Computer

Railroad Command Control Equipment

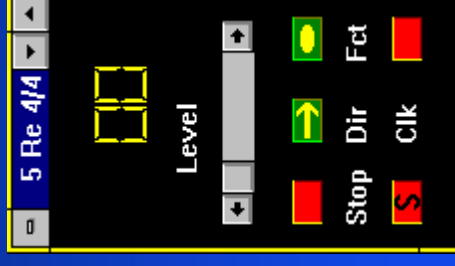
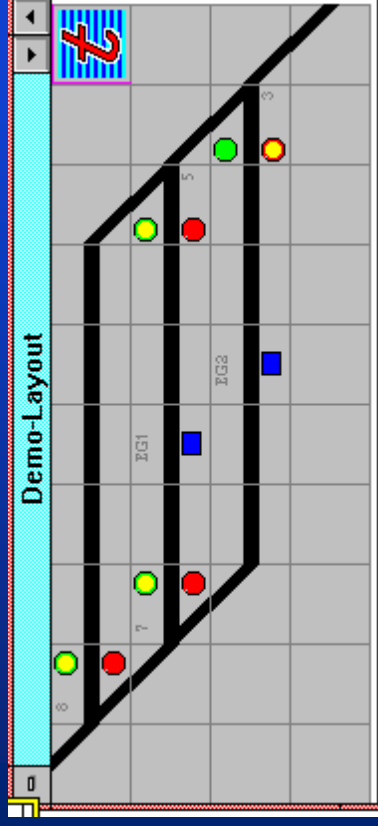
# Command Control Software

- **Three classes of products available**
  - Mac & Apple
  - MS-DOS
  - Microsoft Windows
- **MS-DOS examples**
  - Marklin shareware apps
  - Basic program examples
  - Digipert/Digiplus II
- **Microsoft windows**
  - Engine-Commander
  - WinLok
  - and soon others as well



# WinLok

- Design to support Marklin Controllers
  - draws from the German railways operation
  - supports visual layout display
  - multiple user throttles
  - integrated acceleration curves
- European design/tradeoffs



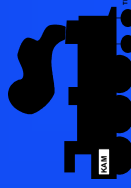
# Engine-Commander™

- **Built on a modular philosophy**
  - users can add complexity at their own pace
  - simulation interface is include to help other programmers
- **Good user documentation**
  - explains the Marklin, Digitrax and Lenz interface in detail
- **Manual and Sensor control flexibility**
  - logical to physical mapping of engines/switches
  - maintains engine and switch history between sessions



# Engine-Commander™ Modes

- **Serial Interfaces supported**
  - Marklin 6050/6023 interfaces on AC and DC boosters
  - Lenz LI-100
  - Digitrax serial Interface planned
  - NMRA serial interface
- **NMRA controllers**
  - Direct drive support for Marklin boosters
  - Direct drive support for NMRA extended packet format
- **Protocol windows for software development**



# Should I write my own programs?

- **Are you ready to**
  - read the protocol specification to the controller
  - write in a computer language
  - spend many hours away from you layout
  - ... have fun programming?
- **What Language do you use?**
  - novice: basic or visual basic
  - experienced: C or Pascal
  - advanced: C++ under windows



**Understand where you want to put your energy  
to maximize your fun!**



# What is the best way to begin?

- **First understand the protocol and interface**
- **Second follow these rules**
  - keep it simple....
  - design the architecture....
  - build the infrastructure....
- **Best way to begin..**
  - buy the correct PC and the Microsoft tools
  - if you are a novice used Visual Basic
  - if you are advance user, use Visual C++



**Remember, Rome was not built in a day!**



# What is the best way to begin?

- **Follow these four steps**
  1. **Acquire a 3rd party app for experimentation**
  2. **Design your user interface (use GUI tool)**
  3. **Now implement small features**
  4. **Add functionality as you desire**
- **Lets walk through these four steps...**

**Remember, Rome was not built in a day!**



# Acquire a 3rd party Application

- Lets look at a demo of Engine-Commander

# Design your GUI Interface

- Visual C++ App builder
- Visual Basic 3.0
  - Sample Applications (VBTerm)
  - drag drop metaphore

# Now Implement small features

- **Make the engine go!**
- **Marklin Interface sample**
  - Start Command
  - Engine Go command

**Remember, Rome was not built in a day!**

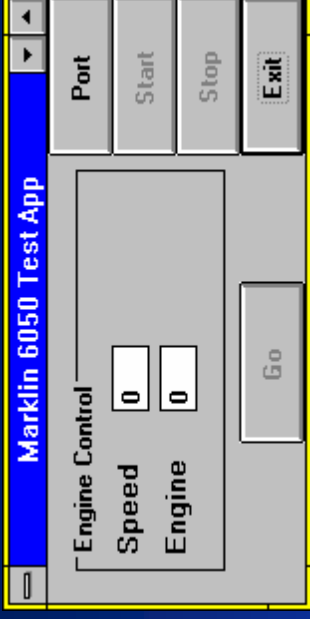


```

// The format of the data packet is
// Format: <engine Address> <direction/Speed>
// Where <Engine Address> is between 1 - 99
// (6050 interface does not transmit address 80 -99)
// <speed/direction> b7 = 0
// b6 = 1
// b5 = Direction 0 for forward, 1 for reverse
// b4 = function set to 1 if on, 0 if off
// b3 - 1 = speed
case ENGINE: {
    BYTE bySpeedDir;
    WORD wEngine;
    LPWORD lpwEng;
    // Get the engine number form the bit sequence
    lpwEng = (LPWORD) (lpSrcData + ENGADDR_PKT); // set the engine peed location
    wEngine = *lpwEng; // get the engine number
    bySpeedDir = lpSrcData[ENGSPPEED_PKT] & 0x0F; // get the speed; 0 - 0xF are speeds
    // Process direction; in this case speed of 0x0F is reverse to the controller
    if (lpSrcData[ENGDIRECT_PKT] bySpeedDir = 0x0F; // set to reverse speed
    // Process function
    if (lpSrcData[ENGFUNC_PKT] bySpeedDir = bySpeedDir | 0x10; // set function on
    else bySpeedDir = bySpeedDir & 0xEF; // set function off (mask to 0)
    lpDstData[0] = bySpeedDir; // Set the speed and engine function
    lpDstData[1] = (BYTE) wEngine; // set the engine address
    iDataSize = 2;}
break;

```





```

Sub cmdGo_Click ()
'--- If the port is opened,
If MSComm1.PortOpen Then
'--- Send the command to the port
MSComm1.Output = Chr$(wEngine) + Chr$(wSpeed)
End If

End Sub

```

# Bottom Line.. Have fun

Lets Look at where to get free  
software!

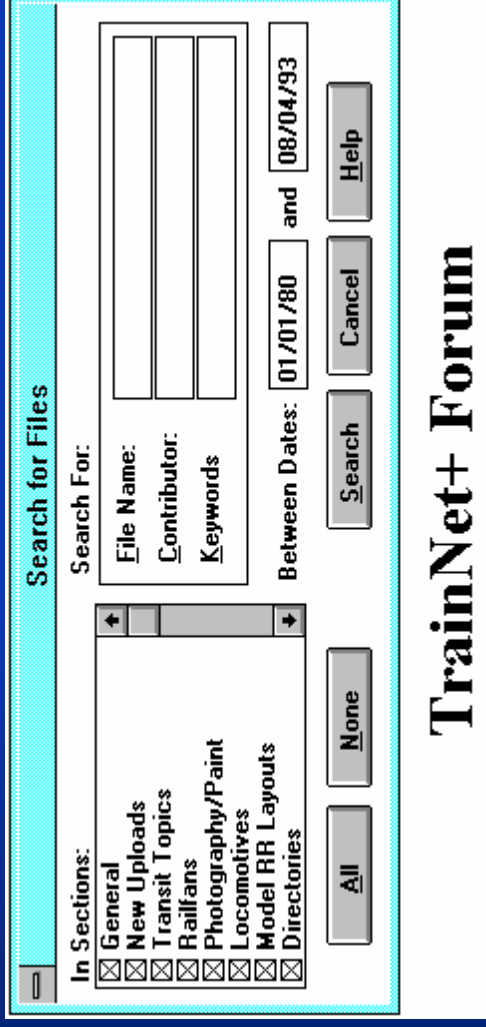


# Shareware Software and Support

- **Compuserve**
  - monthly user fee's
- **The-Conductor BBS**
  - free, just register with the SYSOP
- **Model railroad SIGS**
  - command control SIG
  - computer users SIG
- **Internet**
  - get on Stan Ames internet mailing list

# Compuserve

- Monthly usage fee \$8.95
- Train-net forum is where the action is
- On-line conferences, library, message logs

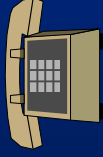




# How to access the BBS

- **Easy to access the BBS**
  - must support ANSI graphics PC users try pro-comm plus or pro-comm for Windows
  - Follow instructions on logon
  - make sure you register on the BBS

- **Two lines for access**



- 503-690-8176 (2400 baud)
- 503-690-4892 (14.4K baud)

- **Lots of demo programs**

- two CDROMS of shareware
- 1.6 Gbyte on line database
- internet access in Q4

## File Library Area

- |     |                           |
|-----|---------------------------|
| [1] | Command & Control Files   |
| [2] | Misc Tools                |
| [3] | Train Photos              |
| [4] | Marklin Library           |
| [5] | Railfan Library           |
| [6] | Train-Basic(TM) Samples   |
| [7] | User File Upload Area     |
| [8] | CICA Windows CD-ROM       |
| [9] | PCSIG CD-ROM ver. 12      |
| [0] | Railroad Simulation Files |

# Questions ?

**Now, lets run Trains!**

