GENNUM CORPORATION

Programmable System Overview GP520A/GP521

INFORMATION NOTE

INTRODUCTION

Rapidly developing technology presents the opportunity to provide better solutions to solving the concerns of the audiological community. The emergence of Gennum's Programmable Series is a prime example of our continuing efforts to deliver effective solutions to customer needs.

Programmability provides avenues for the implementation of new ideas and added flexibility. This flexibility allows for more complex solutions without the mechanical constraints of conventional circuitry. Implementing these complex systems does not necessarily mean more confusion for product users, on the contrary, programmability allows for quick, simple and convenient integration and use of the system.

In this packet, you will find information about Gennum's programmable system. It provides a simple but effective method of implementing the analog and digital functions required of such a system. A description of the programmable chip-set, and necessary hardware and software support is provided in this information note.

The chip-set is composed of the GP520A Analog Signal Path and the GP521 Memory/Controller. Fabrication technology optimal to each application was used. The result is the Bipolar GP520A Analog Signal Path and the CMOS GP521 Memory / Controller.

Adjustment of the GP520A / GP521 chip-set is performed using a programmer. A dedicated programmer which functions specifically to adjust the parameters in the programmable chip set is certainly a possible option, but since personal computers (PC) are so much more convenient and cost-effective, the PC-based programmer route was chosen. The flexibility of the PC allows it to perform not only as a programmer for the chip-set, but also as a word processor, data base, graphic display and a limitless host of other functions. Taking into consideration that many audiometers, real-ear measurement systems, and hearing instrument test boxes are PC-based, it becomes obvious why the PC was selected to perform the programming.

To allow coherent communication between the programmer and the GP520A / GP521 Chip Set, an interface protocol was established. The protocol defines a standard method of communication between the programmer and the Memory/ Controller. In this manner, a family of compatible programmable integrated circuits may be developed. The interface protocol may be made transparent to the end user by means of provided subroutines that take into consideration the correct way to encode the information being communicated. A thorough definition of the protocol is discussed in this information note.

Since the GP520A / GP521 nominally operate at 1.3 V, whereas the computer operates at TTL 5 V levels, an interface is required to satisfy the logic level requirements of each device. The result is the creation of a programmable interface. The interface is designed using off-the-shelf parts and is placed in parallel with the printer. This interface board is totally transparent to the printer and should not affect printer functions. Information regarding the Programmable Interface Module is also included in this information package.

A block diagram of the system is shown in Figure 1.

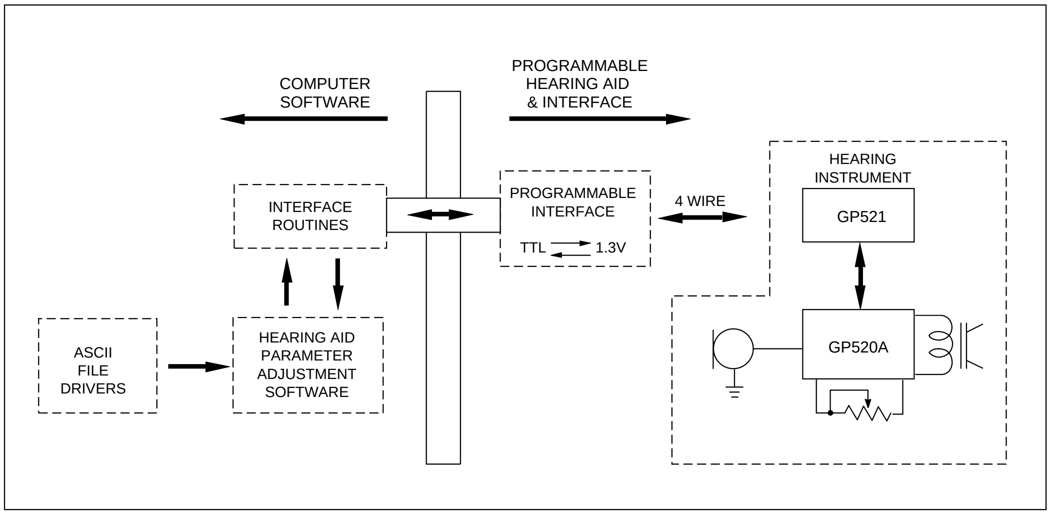

Fig. 1 System Overview

Document No. 520 - 08 - 03

GP520A ANALOG SIGNAL PATH

The GP520A Analog Signal Path has seven adjustable parameters. They are:

- Gain*
- Low Pass Filter
- High Pass Filter
- AGC Threshold
- Release Time
- Maximum Power Output (MPO)
- Receiver Bias Voltage

* Two gain adjustments are provided in the GP520A.

Adjustment of the individual parameters is achieved by mating the GP520A with the current sinks in the GP521 Memory / Controller chip, which may be programmed to step by linear increments. Linear to logarithmic converters are used in the GP520A where appropriate. When used with the GP521, the GP520A supplies a reference current to the GP521 which, through digital control, sets the amount of current it may sink from its eight programmable current sinks (PCS). This in turn sets the parameters in the GP520A. A range of up to 15 increments is possible with the GP521.

The GP520A has both a high-cut and a low-cut 12dB per octave filter block, a 5:1 input compression amplifier, electronically adjustable output level clipping circuit for MPO control, adjustable gain preamplifiers, 2.5 μA reference current source, and a voltage drive class A output stage. The GP 520A also includes an on chip 0.95 V regulator for improved supply rejection.

GP521 CMOS DIGITAL MEMORY / CONTROLLER

The GP521 is designed to control the GP520A Analog Signal Path through the 8 available Programmable Current Sinks (PCS). It uses a combination of temporary RAM and non-volatile EEPROM memory to maintain the correct settings for its current sinks. The non-volatile EEPROMs retain the programmed settings without an available power supply or battery backup. The GP521 also contains digital logic circuitry that performs error-detection, serial to parallel data conversion, and controller functions, as well as updating device status and operations registers.

INTERFACE PROTOCOL

a) HARDWARE

Communication between the programmer and the Memory/ Controller is made serially through a 4 wire cable (3 wires if the Memory/Controller chip and Analog Signal Path chip has its own independent power supply during communication).

The four wire lines are labelled:

- $\overline{\text{DATA}}$
- $\overline{\text{CLOCK}}$
- GROUND
- V_{CC}*

* The Vcc line is not required when the programmable chip-set is already powered by an alternate power source during communication.

The DATA line is bi-directional.

Overline labelling implies that these lines are the complement.

Since the DATA line is bi-directional, it is conceivable that both the programmer and the Memory/Controller circuit may want to send data through the same line. In order to prevent damage to either the programmer or the Memory/Controller chip if this occurs, open collector output with passive pull-up resistor is used at both ends of the communication line. Open collector topologies require that the complement of DATA and CLOCK is sent instead. Pull up resistors for the $\overline{\text{DATA}}$ and $\overline{\text{CLOCK}}$ lines should be placed externally during programming, a value greater than 10 kΩ should be used.

Since fixed baud rate transmission increases circuit complexity, an external clock line supplied by the programmer is needed to provide coherent communication between the programmer and the GP521.

b) DATA FORMAT

The dialogue between the programming unit and the Memory/ Controller chip consists of 16 bits being sent to the Memory/ Controller, which responds by returning an 8 bit word to the programming unit. Interleaved with this data are synchronization bits. The programming unit is required to supply the CLOCK signal throughout the programming procedure.

The format of the signal DATA is defined below:

$f\ a_0\ a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ d_0\ d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ p_0$

The group of bits transmitted by the programmer are defined as follows:

f function bit

$f = 0$ Write Data to specified memory address

$f = 1$ Read Data from specified memory address

$a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0$	7 bit address to read / write to
$d_6\ d_5\ d_4\ d_3\ d_2\ d_1\ d_0$	7 bit data to write to memory
p_0	parity bit information from programmer (odd parity)

Eight bits are sent back by the Memory/Controller circuit in response to the programmer and are in the following format:

$$r_0\ r_1\ r_2\ r_3\ r_4\ r_5\ r_6\ p_1$$

The group of bits transmitted by the Memory/Controller chip are defined as:

$r_6\ r_5\ r_4\ r_3\ r_2\ r_1\ r_0$ 7 bit data sent by Memory/Controller

If f =0 then r_6 to r_0 will echo the 7 bit data d_6 to d_0 sent by the programming unit.

If f =1 then r_6 to r_0 will return the data located at the memory address a_6 to a_0.

p_1 parity bit information from Memory/Controller chip (odd parity)

eg.
function bit - f=0

to send seven bit address - 7 (0000111):
a_6 =0, a_5 =0, a_4 =0, a_3 =0, a_2 =1, a_1 =1, a_0 =1

to send seven bit data - 9 (0001001):
d_6 =0, d_5 =0, d_4 =0, d_3 =1, d_2 =0, d_1 =0, d_0 =1

seven bit data send by controller - 9 (0001001):
r_6 =0, r_5 =0, r_4 =0, r_3 =1, r_2 =0, r_1 =0, r_0 =1

parity bits - p_0=0 p_1=1

However the GP521 controller operates on the principal of the negative logic. This fact is signified by the bar on top of the DATA line - $\overline{\text{DATA}}$.

The actual signal on the data link is $\overline{\text{DATA}}$, the complement of DATA.

A synchronization bit is placed in between each bit of data. The synchronization bit alternates between the two logic states and is generated by the circuit transmitting on the $\overline{\text{DATA}}$ line.

Since the state of every other bit on the data stream is known, error detection may be easily implemented.

It is worthwhile to note that during communication, the least significant bit is sent first.

A complete dialogue between programmer and controller would be:
From the programmer on the $\overline{\text{DATA}}$ line:

$$\overline{f}\ 1\ \overline{a}_0\ 0\ \overline{a}_1\ 1\ \overline{a}_2\ 0\ \overline{a}_3\ 1\ \overline{a}_4\ 0\ \overline{a}_5\ 1\ \overline{a}_6\ 0$$

$$\overline{d}_0\ 1\ \overline{d}_1\ 0\ \overline{d}_2\ 1\ \overline{d}_3\ 0\ \overline{d}_4\ 1\ \overline{d}_5\ 0\ \overline{d}_6\ 1\ \overline{p}_0\ 0$$

From the Controller circuit on the $\overline{\text{DATA}}$ line:

$$\overline{r}_0\ 1\ \overline{r}_1\ 0\ \overline{r}_2\ 1\ \overline{r}_3\ 0\ \overline{r}_4\ 1\ \overline{r}_5\ 0\ \overline{r}_6\ 1\ \overline{p}_1\ 0$$

The relationship between clock and the data stream is shown in the figure below:

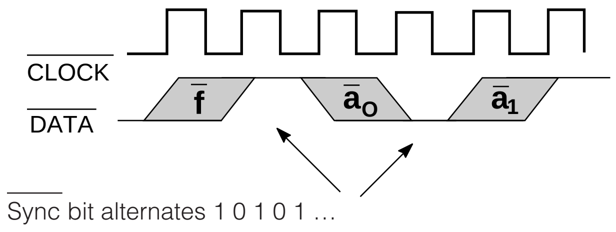
$\overline{\text{CLOCK}}$
$\overline{\text{DATA}}$
$\overline{f}$
$\overline{a}_0$
$\overline{a}_1$
$\overline{\text{Sync}}$ bit alternates 1 0 1 0 1 …

Figure 2

In our example the complete dialogue will be:

from programmer on $\overline{\text{DATA}}$ line
1 1 0 0 0 1 0 0 1 1 1 0 1 1 1 0
0 1 1 0 1 1 0 0 1 1 1 0 1 1 1 0

from GP521 on $\overline{\text{DATA}}$ line
0 1 1 0 1 1 0 0 1 1 1 0 1 1 0 0

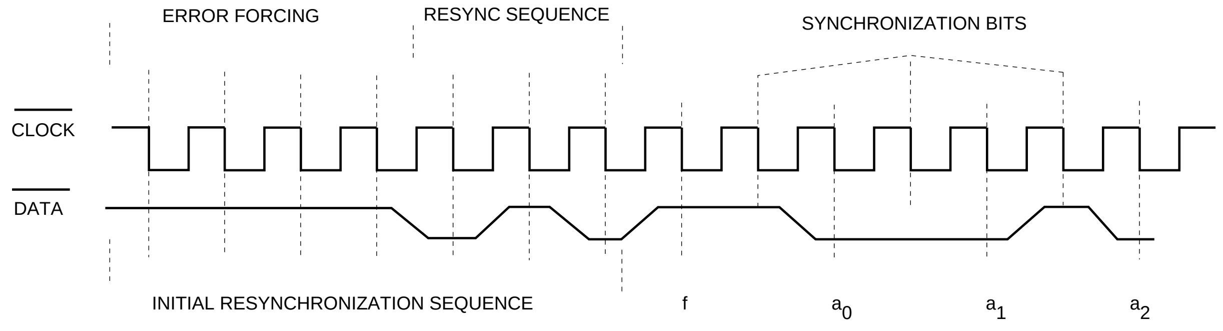

Figure 3 Actual signal on the data link (for example in the text)

For this example the relation between clock and the data for eight resynchronization bits (described in doc. no.510-79) and first four transmission bits from the programmer is shown on Figure 3.

This data format was selected because it allowed several different methods of detecting errors in transmission. The parity bit provides a simple check of data integrity. The parity bit in the return data sent by the Memory/Controller circuit is used as an error flag. If the Memory/Controller circuit detects an error in the data stream, it forces the parity flag to the wrong parity.

Simply relying on parity bits for error detection is not enough. This method detects an odd number of errors only. If an even number of errors occurs during transmission, the parity detected by the recipient will equal the parity sent by the transmitter. Sporadic errors due to faulty cable connections, stuck data lines , and extra or missing clock cycles are some examples of errors that may not be detected from just the parity information. By adding the synchronization bits in the data stream, a stuck data line will be detected by the first or second sync bit. This is possible because the sync bit pattern is already predefined. Deviations from this pattern caused by errors are therefore easily detected. Missing or extra clock cycles may be detected assuming that when the error occurred the erroneous data stream does not mimic the expected data bit pattern. There are a few situations where errors may go undetected.

Errors that are known to escape detection by the Memory/ Controller are listed as follows:

The data line gets stuck at DATA =1 (or an extra clock cycle occurs) at the time p_0 is transmitted and when p_0 is 1. The Memory/Controller will receive the correct data and will perform normally even if the DATA line is stuck. Of course, when the Memory/Controller attempts to write onto the stuck data line, the programmer will not receive the expected data from the Memory/Controller. The interface software on the programmer end should be able to detect the error and correct for it if required.

Extra clock cycles occur with the data correctly mimicking the expected sync bit pattern. This type of error is more likely to go undetected the closer it is to the end of a transmission, since the probability of correctly mimicking the sync bits decreases as the number of bits that is to be mimicked increases. The error will escape detection only when the parity of this bitstream equals the now erroneous parity bit. The probability of such an occurrence is lowered if the *don't care* bits in the data stream are all set at a single value, preventing them from mimicking the sync bit pattern.

Intermittent errors occur on the data line that alter an even number of data bits or, intermittent errors occur on the data line that alter the data bits and the parity bits such that the data stream passes the parity check.

In both cases the assumption is that sync bits have not been altered. Thus, this data stream will pass both the sync bit and parity checks.

Errors may easily be detected by simply comparing the data sent to the Memory/Controller circuit with the data returned. As a last resort, the memory registers in the Memory/Controller circuit may be interrogated and their contents compared to the desired values.

c) REGISTERS

Only one register address is reserved. This is used as an identification register. The address for this register is address 0000000. The register may be made programmable allowing it to be programmed to provide information regarding the hearing instrument (i.e. model number, customer information, serial number etc.) However, general purpose Memory/ Controller circuits, like the GP521, do not have a programmable ID register and thus have their contents set to 0000000, which will allow the chip to be interpreted as not having an ID code. The hearing instrument in this case should be identified by a different method.

Other registers such as a Status Register and Operation Register similar to that of the GP521 may be used. Since the use of such registers are application specific, they are not defined in this standard. See the GP521 Data Sheet for more information regarding these registers.

PROGRAMMABLE INTERFACE

The interface performs the logic conversion that is required when two systems with different logic levels communicate. In other words, the interface converts the 5V TTL logic levels that come from the programmer's parallel port into the 1.3 V logic used in the GP 521 Memory/Controller. Since the DATA line is bi-directional, the interface is also required to convert the 1.3V logic of the GP 521 to TTL levels when the GP521 is communicating to the programmer. To provide electrical isolation from the programmer, the interface uses optocouplers. All parts used in the interface may be easily obtained off the shelf from any of the major electronic component suppliers. A complete schematic and description is given in the information note, 520-9.

SOFTWARE

So far, discussions have been focused on details regarding interfacing and hardware issues. Ultimately, the whole system should be able to interact with the users (Hearing Aid Manufacturers, Audiologists, etc.) who will want to adjust the parameters. Software capable of providing a friendly environment for these users is therefore needed.

This software should be able to convert real world parameters (i.e. gain, high pass corner frequency , etc.) into a binary digital setting sent to the GP521 Memory/Controller.

It should also be able to control an I/O port of the computer and send out data in the form described by the interface protocol.

The flexibility of the PCs may even allow them to obtain data from an audiometer, process the information and follow a fitting procedure to obtain the desired parameter values, as well as providing graphical presentation of the result. Thus, the PC based programmer provides Hearing Aid Manufacturers the opportunity to individualize the parameter adjustment software to fit their own needs and requirements.

In many cases, it will be more beneficial to address the interface protocol concerns through interface protocol subroutines. These subroutines perform the functions of controlling an I/O port in the programmer, as well as the transfer of information as per the interface protocol. Such a subroutine will thus be able to send and receive data in the required format and to monitor and detect errors, making the interface protocol transparent to the actual programming software. In order to facilitate the process of creating the required software, Gennum is offering interface protocol sub-routines that will handle the
I/O functions and ensure that the interface protocol is being adhered to. This set of subroutines is labelled GP521COM and is distributed along with the programmable chip set.

Aside from the GP521COM subroutines, Gennum is also providing two other software items. The first is called TALK521. This software was written as an engineering tool for designers. TALK521 is capable of programming the GP521 Memory/Controller at the digital bitstream level. The other, called USER521, is an example of a user friendly parameter adjustment software.

a) GP521COM

The subroutines in GP521COM are provided to simplify the process of writing programs to communicate with the programmable chip set. Using this approach, parameters are simply passed to the subroutines which then performs the task of sending and receiving data that conforms to the interface protocol. Since it is recommended that the designers of hearing instruments write their own parameter adjustment software, this building block approach of providing subroutines then reduces the development time for writing such programs.

GP521COM is composed of routines that:

- synchronizes the GP521 with the programmer
- sets/resets the parallel printer port of the PC
- sends out the function, address, and data to the memory/controller; receives data from the memory/controller; and relays error messages
- emulates the GP521

These routines may be linked with programs that can be compiled into Microsoft compatible relocatable object codes. Documentation on how this may be done is available with the GP521COM software.

b) TALK521

This program communicates with the GP521 Memory Controller at the digital bitstream level. The program prompts for the function, address, and data. The appropriate data bitstream is then sent to the Memory/Controller. The data sent and received by the programmer and the appropriate error messages are then displayed on the screen. This data is displayed in decimal and binary form. The digital bitstream may be output repetitively while the output of the programmer is monitored with an oscilloscope, providing the user an opportunity to view the actual data stream. As an engineering tool, the program demonstrates how communication using the interface protocol is possible, and provides further insight on what actual signals are being sent by the programmer. The software can also simulate errors that may occur. For evaluation purposes, a GP521 simulator is provided.

c) USER521

USER521 demonstrates a user-friendly programming environment. It describes how the software can read-in a client database, input an audiogram, perform fitting based on an algorithm, and program the GP521. Since this program was written for demonstration purposes only, it does not employ any fitting algorithms. Instead, the software presents possible ideas that may be implemented in order to provide a user-friendly environment.

ASCII driver files, which contain information that maps “real-world” parameters to the appropriate digital settings of the Memory/Controller, are read by USER521. The adjustable “real-world” parameters are displayed on the screen and can be changed using the cursor keys. When connected to the GP520/GP521 chip set, USER521 is capable of adjusting (manually trimming) the various programmable parameters. This option is provided for evaluation purposes only.

HYBRID

The high pin count of both the GP520 and GP521 does not allow for these chips to be incorporated in our traditional packages such as the PLID, MICROpac or SLT. Given the number of external components and the number of interconnections that need to be made, hybrid packaging technology offers a reasonable solution to the needs of such a system. For more information about the programmable hybrid please refer to MAESTRO® I Programmable Hybrid data sheet Document No. 520-41.

Gennum Corporation assumes no responsibility for the use of any circuits described herein and makes no representations that they are free from patent infringement.
© Copyright 1989 Gennum Corporation. Revision Date: March 1992. All rights reserved. Printed in Canada.