Maker Module –

TO12 Interface Definition

# Interface Modes:

The interface between the green bean and the range can take place while the range is in either a Consumer or a Native mode. These modes are defined as follows:

**Consumer Mode**: Allows programmers to access appliance high-level algorithms available to the consumer like initiating a bake cycle at a specified temperature. In **consumer mode** a user connected with the green bean can request a bake cycle to run (or other function available via the front panel) but can not change the low level functions that govern how the cycle runs.

**Native Mode**: When a programmer uses the green bean to connect to the appliance in **native mode**: the API facilitates low-level, direct control of motors, fans, actuators, heaters, and other controlled devices.  High-level algorithms, such as a bake cycle, are not operational.  Native mode allows programmers to, for example, create a new bake cycle by controlling the loads in a desired manner.

# ⚠WARNING:

Range products generate high temperatures necessary to prepare food. While a range operates in Consumer Mode, software applies measures that protect consumers from burn and fire hazards. However, in Native Mode, software controls that are applied through algorithms are not active. Therefore it is the responsibility of the application programmer using Native Mode to follow all guidelines for Safe/Reliable Operation detailed below to protect the consumer from burn and fire hazards.

## Guidelines for Safe/Reliable Native Mode Operation

1. Oven Temperature - oven cavity temperature should not exceed 550 degrees while the door is unlocked. If you intend to run an operation that will exceed cavity temperature of 550 – lock door before starting the cook mode (energizing any elements).
2. Cooling Fan operation – Cooling fan must be on during all cook modes (any heating element energized) and remain on as long as the temperature in either cavity is greater than 175.
3. Surface Temperatures – if operation on any individual oven cavity is to exceed 550 Degrees F -- then the other oven should not be operated during this entire cooking cycle. Concurrent operation of oven cavities at high temperatures (where door lock is required) can overheat components and/or result in high surface temperatures that can injure on contact.
4. Cooktop Operation – do not allow operation of the cooktop if any operating above 550 degrees F is intended.
5. Door Lock operation – the door lock may be incapable of changing state at elevated temperatures. Once a Door is locked, it should remain locked until the cavity temperatures drop below 400 degrees.
6. Open door cooking modes: For models with controls directly above the door – do not allow cook modes to continue with the door open for more than 1 minute as it can result in high component temperatures leading to control damage.
7. Heating Element - A heating element that is on must have its state periodically updated at least every 5 minutes (2.5 minutes is recommended).
8. Relay operation – Excessive relay cycling can shorten the life of the relays.
9. Mode Refresh – The Native Mode must be asserted with the Native Test Mode (0xA0 + 0x01) command at least one time every 5 minutes (2.5 minute periodic rate is recommended).

# Objective:

This documentation is intended to serve as basic instructions for a user interfacing to a TO12 based Range. TO12 controls can operate in either a wall oven configuration or a freestanding electric or gas fueled range + cook top configuration. The maker module documentation related to the interfacing to a specific appliance will focus on the following areas:

1. Interaction Guidelines – guidelines for using the maker module with the TO12 Range products including:
   1. Consumer Mode Interactions – details how to use the product with the currently supported modes to add functionality.
   2. Native Mode Interactions – Details how to use the Native mode to design custom algorithms to add additional functionality to the unit.
2. Command Set – A detailed list of the commands that can be used to interface to the appliance in Consumer and Native modes.

# Interaction Guidelines:

Interaction between an external application and the TO12 via communication will take place in either a normal consumer mode or in a native mode that allows for low-level load control. The mode choice depends upon the degree of control required. If a user wants to extend the existing user interface to more easily control and monitor the behaviors of the oven that already exist, then consumer mode interaction is sufficient. However, if different control algorithms are desired, then the native mode of the control can be used.

## Consumer Mode Interaction:

In the consumer mode it is possible to set and initiate cooking modes and to query for the status while modes are in operation. An external application could use the cooking modes that exist and string them together to create useful compound modes. Example – assume you want to prepare a perfect prime rib recipe (<http://www.simplyrecipes.com/recipes/prime_rib>). It requires 15 minutes at 500 Degrees followed by 14 minutes/lb. at 325 with a recommended termination on a meat thermometer/probe of 115. The ideal way to cook that in a standard oven may not exist without user intervention. Now using a smarter interface you can implement all or most of what is required. Follow these steps for the perfect execution:

1. Enable Remote operation on your oven -
   1. Using either keypad
   2. Or remote command
      1. Use “Upper Oven Remote Enable” ERD (0x510A) – e.g. write ERD and enable
         1. Sample data [Source Address, 0xF1 (ERD write), 0x01 (1 ERD), 0x51, 0x0A (ERD), 0x01 (size), 0x01 (enable)]
2. User your application to Preheat your oven to 500 degrees using “Bake 500”
   1. Either start manually or
   2. Initiate the mode with your application
      1. Use “ERD Write” and “Set Upper Oven Cook Mode ERD” – 0x5100 to select Bake at 500 degrees with no other options
         1. Data = [Source Address, 0xF1 (ERD write), 0x01 (1 ERD), 0x51, 0x00 (ERD), 0x0D (size), 0x01 (bake), 0x01, 0xF4 (500), 0x00, 0x00, (no time terminate), 0x00, 0x00 (no probe temp), 0x00, 0x00 (no delay time), 0x00, 0x00 (no 2 temp cook), 0x00, 0x00 (no 2 temp cook time)
3. Your application can alert you when the preheat process is over and wait for user feedback.
   1. Monitor the cooking status with your application:
      1. Query ERDs “Read Upper Oven State” 0x5101 and Displayed Temperature (0x5109)
         1. Data returned will initially show in Preheat (0x01) and when complete it will show bake 0x05. Assume temperature starts at 100 and rises to 500.
         2. Expected data returned with ERD 5101 (assume only querying x5101 + 5109) = [Source Address, 0xF0 (Query ERD), 0x02 (2 ERDs), 0x51, 0x01 (ERD), 0x01 (data length), 0x01 (preheat), 0x51, 0x09 (ERD), 0x02 (data length), 0x00, 0x64 (100)]
4. While the oven is preheating you prep the roast with your favorite dry rub and insert the meat probe (optional feature)
5. After you application alerts you that preheat is done (cooking state will be bake 0x05) you would place the roast in the oven and plug in the meat probe to oven.
6. Interact with your application letting it know that the meat is in and to complete the cooking process.
7. Application will take it from here
   1. Baking at 500 for 15 minutes
      1. Application starts its 15-minute timer…. And counts down
      2. While counting down the application will use ERDs to monitor the cook state (0x5101, Displayed oven temperature (0x5109), and Probe temperature (0x5103).
      3. Query for all 3 ERDs with one ERD query (0xF0)
      4. Example Response 🡪 [Source Address, 0xF0 (Query ERD), 0x03 (3 ERDs), 0x51, 0x01 (ERD), 0x01 (data length), 0x05 (bake), 0x51, 0x09 (ERD), 0x02 (data length) 0x01, 0xF4 (500), 0x51, 0x03 (ERD), 0x02 (data length), 0x00, 0x48 (temp 72)]
   2. After 15 minutes the application will automatically adjust the cooking temperature to 325 and begin a new timed countdown mode.
      1. Adjustment to Bake 325 is facilitated by altering the cook mode target… Again using “ERD Write” and “Set Upper Oven Cook Mode ERD” – 0x5100 to select Bake at 325 degrees with no other options (assume application monitors the probe!)
         1. Data = [Source Address, 0xF1 (ERD write), 0x01 (1 ERD), 0x51, 0x00 (ERD), 0x0D (size), 0x01 (bake), 0x01, 0x45 (325), 0x00, 0x00, (no time terminate), 0x00, 0x00 (no probe temp), 0x00, 0x00 (no delay time), 0x00, 0x00 (no 2 temp cook), 0x00, 0x00 (no 2 temp cook time)
      2. Assume you are preparing a 10 lb. roast – you want it timed for no more than 2:20 (HH:MM) or when the probe temperature hits 115 whichever comes first….
      3. During count down the application will let you know the time status as well as keep you informed about the Displayed oven temperature and the Probe temperature – assume it queries the same 3 ERDs as earlier for simplicity… return data will look like
         1. 🡪 [Source Address, 0xF0 (Query ERD), 0x03 (3 ERDs), 0x51, 0x01 (ERD), 0x01 (data length), 0x05 (bake), 0x51, 0x09 (ERD), 0x02 (data length) 0x01, 0xF4 (500), 0x51, 0x03 (ERD), 0x02 (data length), 0x00, 0x48 (temp 72)]
   3. Application alerts you when the secondary cook time has hit 2:20 or when the probe temperature hits 115. During the entire process it would monitor the status by:
      1. Feeding back or posting the total cook time
      2. Communicating the current probe temperature. See example above.
   4. If the time is reached first -- an alert is issued… but the oven continues to cook
   5. If the meat probe hits the target
      1. Alert the user
      2. Reduce the cook temperature immediately to 170 (warm)
      3. Use “ERD Write” and “Set Upper Oven Cook Mode ERD” – 0x5100 to select Bake at 500 degrees with no other options
         1. Data = [Source Address, 0xF1 (ERD write), 0x01 (1 ERD), 0x51, 0x00 (ERD), 0x0D (size), 0x12 (warm), 0x00, 0x00 (no temperature), 0x00, 0x00, (no time terminate), 0x00, 0x00 (no probe temp), 0x00, 0x00 (no delay time), 0x00, 0x00 (no 2 temp cook), 0x00, 0x00 (no 2 temp cook time)

The benefit or remaining in consumer mode is that the control is always monitoring the performance and reliable operation of the appliance while still giving the technical user a chance to customize the User Experience as well as create inventive cook modes.

## Native Mode Interaction:

In the Native mode the user/programmer can take a more active role in the control of the oven. However, the reliability and the performance of the oven control is the responsibility of the programmer and their algorithms. The user is capable of individually controlling loads but as a result of this control they are responsible for following the **Guidelines for Safe/Reliable Native Mode Operation.**

Some basic operational techniques described:

1. Locking/unlocking a door – The door lock status can be monitored in consumer mode but it can only be controlled in Native mode. Therefore to lock a door in preparation for higher temperature operation (e.g. >500), entry to Native mode is required. The Lock /Unlock sequence is specifically inhibited by hardware during extreme temperatures to prevent unintended motor operation. Locking and or unlocking the door is achieved by energizing a recirculating DC motor until the sensed state is the intended state. During the transition of the door lock the open/closed state of the door is incapable of being accurately sensed. The process to lock/unlock the door is controlled in all modes of operation. The basic procedure to lock the door is as follows:
   1. Enter the Native mode Using the “Native Mode – Command” (0xA0) – note must be periodically updated or the control will reset and return to consumer mode.
   2. Monitor door Status Lock Status using either:
      1. “Query Discrete Input Status” (0xB1)
      2. Or “Oven Status Update” (0x68)
   3. Trigger a door lock status change using “Control Oven Door Lock” (0xA4)
   4. Continue to monitor the door lock status using commands noted above.
   5. Proceed with Desired behavior after the door lock status has reached the desired state.
2. Baking – In consumer mode this is achieved by use of the Bake and Broil elements. There are cooking phases within the cycle include preheat, Sustain/Bake, and optionally rapid recovery. The Door is monitored and cycling is paused if the door has been open more than 1 minute.
   1. Consumer mode cooking phases are as follows:
      1. Preheat: In the preheat phase the oven cavity temperature is increased rapidly. The Cooking phase transitions from Preheat to Sustain when the temperature in the cavity has reached the target temperature.
      2. Sustain/Bake Mode: Elements are cycled according to a reduced duty cycle profile in order to maintain the oven cavity temperature between hysteresis limits. Cycling of the elements begins based on the temperature error (target temperature - current temperature). Element cycling is enabled when the oven temperature error is >= Hysteresis and continues until the target temperature is reached.
      3. Optional 🡪 Rapid Recovery is enabled when the oven temperature error exceeds a certain value during the Sustain mode (common after door open for a significant period). The object is to increase the cavity temperature quickly before returning to a sustain mode. Phase can be implemented by reentering the preheat phase.
   2. Example of implementing Bake Cycle NATIVE mode:
      1. During cooking maintain several variables as follows:
         1. Target Temperature (Degrees F) – assume 350
         2. Hysteresis (degrees F)
         3. Cooking Mode (Preheat, Sustain, or Terminated) – initial value = Preheat
         4. Temperature Error = (Target Temperature – Current temperature) (units – Degrees F)
         5. Actively Heating flag – True/False – (Initial Value FALSE)
         6. Total Time Counter (seconds) – Initial Value = 0
         7. Element counter (seconds) – initial value = 0
         8. Door Open time counter (seconds) – Initial Value =0
         9. Active Element (Bake/Broil/None) – initial value = None
      2. Phase Variables:
         1. Preheat
            1. Element Sequencing: Element (time in Sec)

Preheat – Bake (80), Broil (45)

* + - * 1. Hysteresis: 0
      1. Sustain/Bake
         1. Element Sequencing: Element (time in Sec)

Preheat – Bake (45), None (45)

* + - * 1. Hysteresis: 5
    1. Basic Bake Cycle Flow
       1. Initialize phase and time variables
       2. Enter NATIVE Mode
          1. Use “Native Mode – Command” (0xA0)
       3. Activate the cooling fan
          1. Use “Control Cooling Fan Speed” (0xAA)
       4. Start Cooking Loop here:
          1. Query current Unit Status included Temperature and Door position.

Temperature

Use “Query Oven Temperatures” (0XAF)

Or -- ERD Query (0xF0) for ERD: “Read Upper/Lower Oven State” (0x5101/0x5201) respectively

Door Position

“Query Discrete Input Status” (0xB1)

Or “Oven Status Update” (0x68)

* + - * 1. Calculate the Temperature Error = (Target Temperature – Actual Temperature)
        2. If (Total Time % 150 == 0) (Checking to see If we need to refresh the NATIVE mode, element, and fan status)

Send Enter NATIVE Mode

Use “Native Mode – Command” (0xA0)

Request the cooling fan ON

Use “Control Cooling Fan Speed” (0xAA)

Reassert the Heating Element Status using Active Element (Bake, Broil, None)

Use “Control Elements Command” (0xA6)

* + - 1. If Cook mode has been terminated:
         1. If the temperature is less than the fan turn off temperature (175) then:

Send message to turn off fan

Use “Control Cooling Fan Speed” (0xAA)

Exit cooking loop

Use “Native Mode – Command” (0xA0) – to terminate the Native mode.

* + - 1. Else If Door Open
         1. Increment the total door open count
      2. Else
         1. Reset door open count to 0
      3. If Door Open time > = 60
         1. Set door open time to 60 (prevent overflow)
         2. Set Active Heating Variable = FALSE
         3. Set active heating element as None
         4. Turn off Heating Elements

Use “Control Elements Command” (0xA6)

* + - 1. Else – if Active Heating == TRUE
         1. If Temperature Error <= 0 (warm enough?)

Set Active Heating to FALSE

Reset the element to first element in the phase 0.

Reset element counter = 0

Set Active Element to None

Turn off Heating Elements

Use “Control Elements Command” (0xA6)

* + - * 1. Else If element counter > element sequence count

Set element counter to 0

Proceed to next element in the cooking profile

Set active element according to the list (will be either Bake, Broil, or None)

Send command to assert the active element.

Use “Control Elements Command” (0xA6)

* + - 1. (Heating active is false) else if Temperature Error > Hysteresis (too Cold)
         1. Set Active Heating = TRUE
         2. Reset the element counter (redundant)
         3. Set Active element to first element in the list
      2. ELSE Active heating is FALSE and to remain FALSE
         1. Turn off Heating Elements (ok to turn off every second)

Use “Control Elements Command” (0xA6)

* + - 1. Wait here until the next second expires.
      2. Repeat Cooking Loop

# Command Set:

Each command that the oven supports will be discussed in general terms. For each command it is important that the user understand the destination address, the command itself (whether it is single or multi-byte), and the additional data that is sent with the command.

Destination Address when dealing with the TO12 device is 0x80.

When a command is transmitted it is sent along with the source address. Source address may not matter and will only be discussed if the command logic takes the source address into account - “Data” area of the command below will generally be talking about what is added after the basic command… Alternate technique is to all out the source address and the command for reinforcement purposes.

The command set is as follows:

1. Read Software Version –
   1. Purpose: Query response to the control to determine current version of the application software. The software version is maintained as AA.BB.CC.DD where each AA-DD are just ## values appears within the range of ’ 00 – 99. The data within the response is represented as a binary/hex value e.g. 99 = 0x63
   2. Command: is equal to 0x01
   3. Data sent with command (beyond source address and command): NA
   4. Example Query
      1. [Source address, 0x01]
   5. Example Response: for version 04.02.03.99 –
      1. [0x80, 0x01, 0x01,0x02,0x03,0x63] where:
         1. 0x80 = Source address of control
         2. 0x01 = command (version response – distinguished from query using total length of packet)
         3. 0x04 = Version AA = 04
         4. 0x02 = Version BB = 02
         5. 0x03 = Version CC = 03
         6. 0x63 = Version DD = 99
2. Native Mode Command/Query
   1. Purpose: This command is used to enter, remain in, or exit the Native mode. The Native mode is used to directly test or control loads within the appliance. Entry to the Native mode is possible only when the appliance is in a standby state where no oven cook modes or other control test modes are active. While in the Native mode the appliance display will show “EOL” or equivalent message. The Native mode is only retained as long as the command to stay in/enter the Native mode is received every 5 minutes or faster. When intending to stay in this mode it is recommended that the command is issued every 2.5 minutes. Failure to refresh the Native mode will reset in a control reset and a return to the consumer mode. Each time the command is issued to enter or continue the Native Mode mode the control will respond with a confirmation message indicating the operational mode. If the command is issued to exit the Native mode the control is reset and no confirmation response is issued.
   2. Command: 0xA0
   3. Data sent with command (beyond source address and command): 1 byte indicated mode command type where:
      1. 0x01 = Enter/Remain in - Native Mode.
      2. 0x00 = exit Native Mode
      3. Other = Query to ask what current mode is (useful for verifying consumer mode)
   4. Example commands:
      1. Enter/Remain in Native mode – [Source Address, Command, 0x01]
      2. Exit Native Mode - [Source Address, Command, 0x00]
      3. Query Current Mode - [Source Address, Command, 0x##] - where 0x## is neither 0x00 or 0x01.
   5. Example response (only provided for a query of mode or Enter Native mode.
      1. [0x80,0x##] where
         1. 0x80 = Source address of control
         2. 0x## will be:
            1. 0x00 – Consumer mode.
            2. 0x01 = Native mode
            3. 0x02 = Service Mode
            4. 0x03 = Native mode
            5. 0x04 = Native mode via keypress
            6. 0x05 = F-Code retrieval mode.
3. Query Key Status (0xB3) –
   1. Purpose: Command is normally used in test modes to monitor the operation of the keyboard. It is intended to return a list of the current keys that are pressed, or all keys that have been pressed since the key latch was last reset.
   2. Command: 0xB3
   3. Data sent with command (beyond source address and command): single byte used to clarify the request. Byte clarifies the query as follows:
      1. 0x00 = read the keys currently pressed (1/0 = pressed/released)
      2. 0x01 = return the key latch value (all keys that have been pressed since the last reset.
      3. 0x02 (or any other value) = Clear the key latch value
   4. Response: Dependent upon the query that was sent but the response will be directed to the broadcast address and will include 6 bytes with keys mapped to each bit value. The data will represent either the current status of the key or the latched value.
      1. For read the keys currently pressed query (0x00) the data will represent the current key state. The keys are mapped dynamically by model.
      2. For return the key latch value query (0x01) – same data as in key query – however the data represents whether or not the key has been pressed since the value was reset not if it is currently pressed.
      3. For clear the latch value query (0x02) – same data as the prior key latch request. However all 5 bytes should now be 0 since the latch was just reset.
   5. Example Command (in the data area only)
      1. [Source address, 0xB3, 0x##] where ## is from 0x00 – 0x02.
   6. Example Response – all example responses look similar – the data will represent the current state or the latch value. Note that regardless of the source address the response is directed to the broadcast address of 0xF4.
      1. [0x80,0x##, 0x01,0x02,0x03,0x04,0x05,0x06] where
         1. 0x80 = Source address of control
         2. 0x## = 0x00 – 0x02
         3. 0x01 – 0x06 = 6 bytes representing the key status. Bit set indicates pressed or was pressed.
4. Test Mode Control LEDs Command
   1. Purpose: Used in test modes to control the data that is posted on all of the LEDs in the UI. The command includes 26 bytes that map all of the individual LEDs in the system. The first 13 Bytes are associated with the upper oven and the 2nd group of 13 bytes is associated with the Lower oven. Setting a bit would light the LED mapped to that bit. Clearing the bit would turn off the LED. Command is ignored in consumer modes.
      1. Seven Segment Display Characters are mapped as follows:

a a a a a a

c b

c b

d d d d d d

e f

e f

g g g g g g

* + 1. Bit positions for each character are mapped in order with bit 0 mapped to segment a and bit 6 of the byte mapped to segment g. e.g. to display a 4’4’ you would send 0B0010 1110 = 0x2e (lighting segments f,d,c, and b).
    2. Each oven has 3 temperature displays and for time displays as well as 6 additional bytes covering the discrete LEDs associated with that oven.
  1. Command: 0xB4
  2. Data sent with command (beyond source address and command): 26 Bytes -- The bytes are mapped to LEDs as follows:
     1. Byte 1 – Upper Oven 1st 7-segment display (e.g. if temp were 350 – this would be the mapping that represents a ‘3’
     2. Byte 2 – Upper Oven 2nd 7-segment display (e.g. if temp were 350 – this would be the ‘5’
     3. Byte 3 – Upper Oven 3rd 7-segment display (e.g. if temp were 350 – this would be the ‘0’
     4. Byte 4 – Upper Oven 1st 7 segment display character in the time area. (E.g. if time were: 04:59 p.m. this would be the ‘0’ character)
     5. Byte 5 – Upper Oven 2nd 7 segment display character in the time area. (E.g. if time were: 04:59 p.m. this would be the ‘4’ character)
     6. Byte 6 – Upper Oven 3rd 7 segment display character in the time area. (E.g. if time were: 04:59 p.m. this would be the ‘5’ character)
     7. Byte 7 – Upper Oven 4th 7 segment display character in the time area. (E.g. if time were: 04:59 p.m. this would be the ‘9’ character)
     8. F Byte 8 – bit mapped to discrete Upper Oven LEDs as follows:
        1. Bit 0 – none
        2. Bit 1 - colon
        3. Bit 2 – none
        4. Bit 3 – remote icon
        5. Bit 4 – remote
        6. Bit 5 – remote
        7. Bit 6 – timer
        8. Bit 7 – none
     9. Byte 9 – bit mapped to discrete Upper Oven LEDs as follows:
        1. Bit 7 – none
        2. Bit 6 – none
        3. Bit 5 – Controls Right
        4. Bit 4 – Controls left
        5. Bit 3 – “Locked”
        6. Bit 2 – “Door”
        7. Bit 1 - “Clean”
        8. Bit 0 – “Self”
     10. Byte 10 – bit mapped to discrete Upper Oven LEDs as follows:
         1. Bit 7 – none
         2. Bit 6 – “AM”
         3. Bit 5 – “PM”
         4. Bit 4 – “Delay”
         5. Bit 3 – “Clock”
         6. Bit 2 – “Time”
         7. Bit 1 - “Cook”
         8. Bit 0 – “Steam”
     11. Byte 11 – bit mapped to discrete Upper Oven LEDs as follows:
         1. Bit 7 – none
         2. Bit 6 – “Rack”
         3. Bit 5 – “Multi”
         4. Bit 4 – “Warm”
         5. Bit 3 – “Proof”
         6. Bit 2 – “Crisp”
         7. Bit 1 - “Roast”
         8. Bit 0 – “Probe”
     12. Byte 12 – bit mapped to discrete Upper Oven LEDs as follows:
         1. Bit 7 – none
         2. Bit 6 – “Fahrenheit”
         3. Bit 5 – “Celsius”
         4. Bit 4 – “Pre”
         5. Bit 3 – “Broil”
         6. Bit 2 – “Bake”
         7. Bit 1 - “Convection”
         8. Bit 0 – “Convection”
     13. Byte 13 – bit mapped to discrete Upper Oven LEDs as follows:
         1. Bit 7 – none
         2. Bit 6 – none
         3. Bit 5 – none
         4. Bit 4 – none
         5. Bit 3 – none
         6. Bit 2 – none
         7. Bit 1 - “Upper Oven” – right side of icon
         8. Bit 0 – “Upper Oven” left side of icon
     14. Bytes 14 – 16 – Temperature display associated with Lower oven
     15. Bytes 17 – 20 – unused
     16. Bytes 21-26 – mapped to same LEDs as bytes 8-13 but associated with lower oven instead of upper oven.
  3. Response: NA – no response from the control
  4. Example Command – not included at this time.
  5. Example Response - NA

1. Control Buzzer Command
   1. Purpose: Command is used during a test mode to play a tone sequence on the Buzzer.
   2. Command: 0xB5
   3. Data sent with command (beyond source address and command): Single byte corresponding with the tone to be played. Tone number ranges from 0 to 20. Common tones used include:
      1. 0 – Buzzer off
      2. 2 – Tone on a valid touch/keypress
      3. 3 – Tone on an invalid touch/keypress
      4. 4 - Tone issued on an error in operation (similar to 3)
      5. 18 – end of cycle
   4. Response: NA
   5. Example Command
      1. [Source address, 0xB5, ##] – where ## is form 0 to 20 or 0x00 to 0x14
   6. Example Response - NA
2. Control Elements Command
   1. Purpose: Allows user to control individual heating elements during native mode. (See Native Mode Command/Query above). In order to maintain the status of the relays the Control Elements command must be repeated or refreshed at an interval no more than 5 minutes.
   2. Command: 0xA6
   3. Data sent with command (beyond source address and command): 2 element control bytes for Upper and lower ovens respectively. If only 1 oven only the upper byte matters. The control byte is bit mapped as follows:
      1. Bit 7-6 = unused.
      2. Bit 5 = unused ( convection 2 Relay in other systems)
      3. Bit 4 = convection 1 Relay
      4. Bit 3 = unused (Broil Relay 2 in other systems)
      5. Bit 2 = Broil Relay 1
      6. Bit 1 = unused (Bake 2 relay in other systems)
      7. Bit 0 = Bake Relay 1
   4. Response: NA
   5. Example Command: Command issued to turn the Bake element on for the upper oven and the broil element on for the lower oven.
      1. [Source address, 0xA6, 0x01,0x04]
   6. Example Response - NA
3. Element Status Query
   1. Purpose: Query for the current state of the heating elements.
   2. Command: 0xA7
   3. Data sent with command (beyond source address and command): NA
   4. Response: Contains bit encoded data representing the status of the load. 1 = ON. Response has 2 key data bytes that are geared toward the loads in the upper and lower ovens. Both bytes are bit coded for the respective ovens as follows:
      1. Bit 7-6 = unused
      2. Bit 5 = unused ( Convection 1 Relay in other systems)
      3. Bit 4 = Convection 1 Relay
      4. Bit 3 = unused (Broil 1 relay in other systems)
      5. Bit 2 = Broil 1 relay
      6. Bit 1 = unused (Bake 2 relay in other systems)
      7. Bit 0 = Bake 1 Relay
   5. Example Command:
      1. [Source address, 0xA7]
   6. Example Response: Representing Bake relay/element on in the Upper oven and the broil relay/element on in the lower oven.
      1. [0x80, 0xA7, 0x01, 0x04]
4. Convection Fan Control
   1. Purpose: Used during testing modes to activate the convection fan in the upper or lower oven at a given duty cycle and in a given direction. Most ovens support at most 1 convection fan. Not all ovens if they have a fan support the ability to change direction nor do they have ability to adjust duty cycle beyond 0 and 100%.
   2. Command: 0xA8
   3. Data sent with command (beyond source address and command): 4 bytes that encode the duty cycle (0-100 %) and the direction (1 = CW, 0 = CCW) for each convection fan. The data is arranged with the upper oven fan noted first (duty cycle and direction) followed by the lower oven.
   4. Response: NA
   5. Example Command: Setting the convection fan in the upper oven on (100% duty) in a CW direction.
      1. [0x80, 0xA8, 0x64, 0x01, 0x00, 0x00]
   6. Example Response: NA
5. Query Convection Fan Status
   1. Purpose: Query the current status of the convection fan.
   2. Command: 0xA9
   3. Data sent with command (beyond source address and command): NA
   4. Response: 4 bytes that encode the duty cycle (0-100%) and the direction (1 = CW, 0 = CCW) for each convection fan. The data is arranged with the upper oven fan noted first (duty cycle and direction) followed by the lower oven.
   5. Example Command: [0x80, 0xA9]
   6. Example Response: (indicating that the Upper oven fan is on in the CW direction
      1. [Querying Address, 0xA9, 0x64, 0x01, 0x00, 0x00]
6. Control Cooling Fan Speed
   1. Purpose: used during testing modes to control the speed of the cooling fan. The cooling fan is used to maintain operational temperature of the electronics and the oven chassis.
   2. Command: 0xAA
   3. Data sent with command (beyond source address and command): 2 bytes that are intended for the control of up to 2 different fans (upper and lower oven). The bytes are encoded as 0, 1, and 2 for speed settings of off, low and high respectively. In most units only the upper fan is populated. Likewise in most models the fan has only 1 speed setting as a result only the Low value and off values should be used in that configuration. Unit will only respond to a turn off command if the oven temperatures are currently below some parametric value that is model dependent. Normally the value is set to approximately 200 Degrees F.
   4. Response: NA
   5. Example Command: Turning the fan on.
      1. [0x80, 0xAA, 0x01,0x00]
   6. Example Response: NA
7. Query Cooling Fan Speed
   1. Purpose: Query the status of the cooling fans in the unit.
   2. Command: 0xAB
   3. Data sent with command (beyond source address and command):
   4. Response: ): 2 bytes that are intended for the control of up to 2 different fans (upper and lower oven). The bytes are encoded as 0, 1, and 2 for speed settings of off, low and high respectively.
   5. Example Command: [0x80, 0xAB]
   6. Example Response: returning status that cooling fan is on.
      1. [Querying Address, , 0xAB, 0x01,0x00]
8. Query Cooling Fan RPM
   1. Purpose: Intended to query the current RPM of the cooling fan. However the RPM is not measured in all controls. In that case only the status is returned.
   2. Command: 0xAC
   3. Data sent with command (beyond source address and command): NA
   4. Response: If RPM is measured then 2 bytes each arranged MSB, LSB for the upper and lower cooling fan RPMs. Note only 1 fan is normally present and in that case the RPM is populated in the upper cooling fan location. When RPM is not measured only 2 bytes may be returned indicating on or off for the Upper and low fans respectively.
   5. Example Command: [0x80, 0xAC]
   6. Example Response:
      1. Assuming RPM is measured 🡪 Cooling fan at 1000 RPM (0x03E8)
         1. [Querying Address, , 0xAC, 0x03,0xE8,0x00,0x00]
      2. If RPM is not measured - -return for cooling fan on at 1000 rpm
         1. [Querying Address, , 0xAC, 0x01,0x00]
9. Control Oven Door Lock
   1. Purpose: Used in test modes to control the oven door locks.
   2. Command: 0xA4
   3. Data sent with command (beyond source address and command): Lock action command (1 = lock, 0 = unlock) sent in 2 bytes for the upper and lower oven respectively. Note in some models when the either door is locked both doors are locked. In this case the door lock mechanism is tied to the upper oven control.
   4. Response: NA
   5. Example Command: Command to lock upper oven.
      1. [0x80, 0xA4, 0x01, 0x00]
   6. Example Response: NA
10. Query Oven Temperatures
    1. Purpose: Used to read the current oven temperature readings.
    2. Command: 0xAF
    3. Data sent with command (beyond source address and command): NA
    4. Response: The temperatures are returned as 3 different temperatures. 2 Temperatures representing cavity temperature readings (fine and coarse respectively) and the temperature for the probe. All temperatures are returned as 16-bit quantities (MSB, LSB) in degrees F.
    5. Example Command:
       1. [0x80, 0xAF]
    6. Example Response: Assumes the oven temperatures for the upper oven are 290 (0x0122), 300(0x012c)), 165 (0x00A5) for the fine cavity, coarse cavity, and probe temperature respectively. Lower oven temperatures not populated.
       1. [Querying Address, , 0xAF, 0x01,0x22, 0x01, 0x2c, 0x00, 0xA5]
11. Query Discrete Input Status
    1. Purpose: Query the current status of various inputs in the unit.
    2. Command: 0xB1
    3. Data sent with command (beyond source address and command): NA
    4. Response: Response includes 2 bytes for each oven (upper and lower ovens respectively). Each of the bits in the return values are bit encoded. As follows:
       1. Byte 0
          1. Bit 7 = Probe Connection status (1/0 = connected?)
          2. Bit 6 = convection Fan Direction (1 = CW, 0 = CCW)
          3. Bit 5-4 = Oven Light Status (11/10/01/00 = NA/On/Off/NA)
          4. Bit 3-2 = Door lock status (11/10/01/00 = Transition/Locked/Unlocked/Unknown)
          5. Bit 1-0 = Door Switch status (11/10/01/00 = NA/Closed/Open/Unknown)
       2. Byte 1
          1. Bit 7 = Fault Status
          2. Bit 6 = Debug Led
          3. Bit 5-4 = AC Frequency (AC Frequency detected – 11/10/01/00 = NA/60 Hz/50Hz/Invalid)
          4. Bit 3-2 = AC Voltage populated in some models but in these models always 0.
          5. Bit 1 = Door initialized – (1/0 = is door initialization complete)
          6. Bit 0 = Buzzer Type (continuous or 1 time)
    5. Example Command:
       1. [0x80, 0xB1]
    6. Example Response: example given shows Upper oven door as open, and door initialization complete, lower oven door as open, and line frequency at 60 hz.
       1. [Querying Address, , 0xB1, 0x01,0x02,0x04,0x20]
12. Query Cooking Mode Status
    1. Purpose: Determine the cooking mode status for each of the ovens.
    2. Command: 0xAE
    3. Data sent with command (beyond source address and command): NA
    4. Response: The return value for each oven (upper and lower respectively) includes the cook mode and a 32 bit quantity representing the error status. At this time the error status is always zero for this control. Not all modes are supported in each model. Cook mode is coded as follows:
       1. 0 = Inactive
       2. 1 = Bake Mode no option
       3. 2 = Probe Bake
       4. 3 = Delay Start Bake
       5. 4 = Delay Start Probe Bake
       6. 5 = Bake Cook and Hold
       7. 6 = 2 Temp Bake
       8. 7 = Cook and hold 2 temp bake
       9. 8 = Broil Low
       10. 9 = broil low with probe
       11. 10 = broil high
       12. 11 = broil high with probe
       13. 12 = Convection Bake No options
       14. 13 = convection bake with probe
       15. 14 = convection bake with delay start
       16. 15 = convection bake with probe delay start
       17. 16 = convection bake timed shutoff
       18. 17 = convection bake timed shutoff, delayed start
       19. 18 = convection multi-bake
       20. 19 = convection multi-bake with probe
       21. 20 = convection multi-bake with delayed start
       22. 21 = convection multi-bake with probe delayed start
       23. 22 = convection multi-bake with timed shutoff
       24. 23 = convection multi-bake with timed shutoff delayed start
       25. 24 = convection roast
       26. 25 = convection roast probe
       27. 26 = convection roast delayed start
       28. 27 = convection roast probe delayed start
       29. 28 = convection roast timed shutoff
       30. 29 = convection roast timed shutoff delayed start
       31. 30 = self-clean
       32. 31 = self-clean delay start
       33. 32 = steam clean
       34. 33 = warm
       35. 34 = slow cook
       36. 35 = pizza
       37. 36 = proof
    5. Example Command:
       1. [0x80, 0xAE]
    6. Example Response: Response for a Bake in both ovens…
       1. [Querying Address, , 0xAE, 0x01,0x00,0x000x00,0x00, 0x01,0x00,0x000x00,0x00]
13. Oven Status Update
    1. Purpose: Command issued to query the full status of either the upper or the lower oven.
    2. Command: 0x68
    3. Data sent with command (beyond source address and command): Single byte indicating which oven status you are querying. (0 = Upper, 1 = Lower oven).
    4. Response: 26 byte response (including the querying address and command), that is coded similar to other queries discussed above. The response includes a status snapshot for the operation of an oven cavity. It includes the following:
       1. Querying Address
       2. Command - 0x68
       3. Oven - (0 = Upper, 1 = Lower)
       4. Cook mode - -coded identically to 0xAE response
       5. Cook Primitive – coded value indicating the cooking stage (preheat etc… )
       6. LSB of current oven temp in Degrees F
       7. MSB of current oven temp in Degrees F
       8. LSB of the Set point in Degrees F
       9. MSB of Set point in Degrees F
       10. Heating element Status (coded the same as similar information to but coded differently than Element Status Query (0xA7) discussed above:
       11. LSB of Generic Relay Status
           1. Bit 7- Convection Fan On Status
           2. Bit 6-Oven Light Status
           3. Bit 5-Oven Light Status
           4. Bit 4-Door Lock Status
           5. Bit 3-Door Lock Status
           6. Bit 2-Door Lock Status
           7. Bit 1-Door switch
           8. Bit 0-Door switch
       12. MSB of generic relay status
           1. Bit 7/15-Not Used
           2. 14-Not Used
           3. 13-Not Used
           4. 12-Not Used
           5. 11-Not Used
           6. 10-FCODE
           7. 9-ProbeConnected
           8. 8-FanDirection
       13. LSB of ADC channel 0
       14. MSB of ADC channel 0
       15. LSB of ADC channel 1
       16. MSB of ADC channel 1
       17. LSB of ADC channel 2
       18. MSB of ADC channel 2
       19. LSB of ADC channel 3
       20. MSB of ADC channel 3
       21. Unused byte
       22. Unused byte
       23. Oven Temperature LSB – Degrees F
       24. Oven Temperature MSB – Degrees F
    5. Example Command: (query for status of the upper oven.
       1. [0x80, 0x68, 0x00]
    6. Example Response: Full sample not presented - -shown is partial for upper oven status read.
       1. [Querying Address, , 0xAC, 0x01,0x00…]
14. Parametric Data Record Query
    1. Purpose: Query of start address of parametric data structure. Used mainly for autopsy/debug purposes and a read out of parametric data.
    2. Command: 0x30
    3. Data sent with command (beyond source address and command): NA
    4. Response: 32-bit address of the start of the parametric data record.
    5. Example Command:
       1. [0x80, 0x30]
    6. Example Response: Assuming the parametric data started at address 0x01020304
       1. [Querying Address, , 0x30, 0x01, 0x02, 0x03, 0x04]
15. U8\_CMD\_RESET\_EEPROM\_TO\_DEFAULTS
    1. Purpose: Command issued only in end of line mode that is used to reset non-volatile data storage to default/factory values.
    2. Command: 0x34
    3. Data sent with command (beyond source address and command): NA
    4. Response: If command is executed a response command will be returned to confirm.
    5. Example Command:
       1. [0x80, 0x34]
    6. Example Response: Fixed response if the command is executed as follows:
       1. [Querying Address, , 0x34 0x01]
16. Read Glass Touch Error Status
    1. Purpose: Reads the current value for the Glass touch error status. Model dependent results.
    2. Command: 0x15
    3. Data sent with command (beyond source address and command): NA
    4. Response: 3 bytes indicating the status of the glass touch chip interface:
       1. Byte 1 - bit coded status
          1. Bit 7 – b1LEdOn
          2. Bit 6 - Communication Failure
          3. Bit 5 – FMEA Failure
          4. Bit 4 – EEPROM Corrupt
          5. Bit 3 – SCLK Failure
          6. Bit 2 – Calibration Fail (of cap touch control)
          7. Bit 1 – Key Calibrate
          8. Bit 0 – Key Detect
       2. Bytes 2,3 –form 16-bit CRC value
    5. Example Command:
       1. [0x80, 0x15]
    6. Example Response: assumed no failures CRC 0x0001
       1. [Querying Address, 0x15, 0x00,0x00,0x01]
17. Toggle Oven Light
    1. Purpose: Command sent to toggle the current status of the oven light.
    2. Command: 0x71
    3. Data sent with command (beyond source address and command): NA
    4. Response: NA
    5. Example Command:
       1. [0x80, 0x71]
    6. Example Response: NA
18. Get Fault Snapshot Header
    1. Purpose: Query to gain information related to the fault snapshot.
    2. Command: 0x92
    3. Data sent with command (beyond source address and command): NA
    4. Response: Response includes information used to read the fault snapshot. Specific data returned as follows:
       1. # of fault snapshots stored
       2. # of queries/pages used to read an individual snapshot
       3. size of an individual fault snapshot
       4. # of entries in the cycle history
       5. # of queries/pages used to read an entry in the cycle history
       6. Size of the individual cycle history entry
    5. Example Command:
       1. [0x80, 0x92]
    6. Example Response: Typical example has 2 fault snapshots,2 queries to read a snapshot, 44 bytes per entry, 2 entries in cycle history and 2 queries to read cycle history entry, 81 bytes for an individual cycle entry.
       1. [Querying Address, 0x92, 0x02,0x02,0x2C, 0x02, 0x02, 0x51]
19. Request Fault Snapshot Entry
    1. Purpose: Read an individual page from the fault snapshot.
    2. Command: 0x89
    3. Data sent with command (beyond source address and command):
       1. Snapshot Index
       2. Page of fault snapshot.
    4. Response: Requested data – parsing to take place by requester. To help parse the response the return value includes the snapshot number and snapshot page with the data.
    5. Example Command: Asking for page 1 of fault 2.
       1. [0x80, 0x89,0x02,0x01]
    6. Example Response: Partial example… to request for page 1 of fault 2….
       1. [Querying Address, 0x89, 0x02,0x01, ….]
20. Request Cycle History Entry
    1. Purpose: Read an individual page from the cycle history.
    2. Command: 0x88
    3. Data sent with command (beyond source address and command):
       1. Cycle history Index
       2. Page of cycle history
    4. Response: Requested data – parsing to take place by requester. To help parse the response the return value includes the cycle history number and page with the data.
    5. Example Command:
       1. [0x800x88,0x02,0x01]
    6. Example Response:
       1. [Querying Address, 0x88, 0x02,0x01…]
21. Request FCODE Data Format
    1. Purpose: Query for format version history for the F-Code. Used by requestor to assist in parsing F-Code data.
    2. Command: 0x85
    3. Data sent with command (beyond source address and command): NA
    4. Response: Includes:
       1. Major FCODE Format version
       2. Minor FCODE Format Version
       3. 0x00
       4. 0x00
       5. Fault Count
    5. Example Command:
       1. [0x80, 0x85]
    6. Example Response: Typical response with current version and no F Codes at this time.
       1. [Querying Address, 0x85, 0x01,0x01,0x00,0x00,0x00]
22. Request FCODE
    1. Purpose: Read the value stored in the FCODE buffer.
    2. Command: 0x86
    3. Data sent with command (beyond source address and command): 1 byte that designates which fault code is to be read. Note if the index is out of bounds for the fault codes available the last fault code will be returned.
    4. Response: The fault code of interest is returned. Data that is part of the fault code includes:
       1. Index of fault number
       2. LSB of Fault code Number
       3. MSB of fault code number
       4. 4 byte quantity that defines the relative time (in minutes) the fault code took place. This is sent in big endian.
       5. U8 – count of how many times the FCODE has happened
       6. U8 – indication of whether the FCODE is currently active or not. (1 = active)
    5. Example Command: Query for FCODE at offset #1
       1. [0x80, 0x86,0x01]
    6. Example Response: 0x01 indexes FCODE returned… example is fault number 4 (0x0004), occurs at relative time of 3241243 minutes 0x0031751B, occurred 2 times, currently inactive.
       1. [Querying Address, , 0x86, 0x01,0x04, 0x00, 0x00, 0x31, 0x75, 0x1B,0x02,0x00]
23. Clear FCODE Request
    1. Purpose: Clears all system errors and FCODEs including counts.
    2. Command: 0x87
    3. Data sent with command (beyond source address and command): NA
    4. Response: NA
    5. Example Command:
       1. [0x80, 0x87]
    6. Example Response: NA
24. ERD Query
    1. Purpose: Query for Entity Reference Designator Data. This command is similar to a read. Requestor can ask for the data associated with 1 or more ERDs.
    2. Command: 0xF0
    3. Data sent with command (beyond source address and command): data that clarifies which ERDs are being queried.
       1. N - 1 byte representing the number of ERDs being requested
       2. N - 16- bit in big endian format that define the ERDs being requested.
    4. Response: The ERDs that were requested are returned assuming they fit in a transmittable data packet. The return format after the command and how many ERDs is noted is an array of ERDs and their associated data.
    5. Example Command: Example is a request for the Model number ERD.
       1. [0x80, 0x F0,0x01 0x00,0x01
    6. Example Response: Example is a request for the Model number ERD. The format of the response is the number of ERDs included in the response followed by the array of erd data. Each ERD entry includes the 16- bit ERD number (in this example 0x0001), followed by a byte indicating the size of the ERD data (32 bytes in this example), followed by the data (not shown).
       1. [0x80, 0xF0, 0x01, 0x00, 0x01, 0x20,…]
25. ERD Write
    1. Purpose: Writing values to designated ERDs
    2. Command: 0xF1
    3. Data sent with command (beyond source address and command): N # of ERDs and N ERD structures (ERD #, size and data…). Everything in big endian
       1. N = 1 byte - # of ERDs that are being written
       2. N – ERD structures…
          1. 16- bit related to the ERD to write
          2. Y - 8-bit – size of ERD to write
          3. Y – bytes Data to be written
    4. Response: Return value includes a count of and a list of how many ERDs were actually written.
       1. N = 8- bit that represents the count
       2. N – 16-bit ERD #s that confirm the ERD was written.
    5. Example Command: Example is a request for the Model number ERD.
       1. [0x80, 0x F1, 0x01 0x00,0x01,0x20,Data….]
    6. Example Response: NA
       1. [0x80, 0xF1, 0x01 0x00,0x010]
26. ERD Subscribe
    1. Purpose: Gives ability for an entity to be alerted to the status of certain ERD’s either periodically or when the value changes.
    2. Command: 0xF2
    3. Data sent with command (beyond source address and command): The data included is the number of ERDs that are being subscribed to and a structure related to the subscription for each of those ERDs.
       1. N = # of ERDs that are being subscribed to.
       2. N- Structures related to the subscription including:
          1. 16- bit ERD #
          2. Byte – subscription time or 0 for alert on change. Note only alert on change is currently supported.
    4. Response: A count of the number of ERD’s that were subscribed to. Single byte
    5. Example Command: Example is a request for the Model number ERD – transmit on change.
       1. [0x80, 0x F2,0x01 0x00,0x01,0x00]
    6. Example Response: 1 ERD subscribed.
       1. [0x80, 0xF2, 0x01]
27. Request current Subscription List
    1. Purpose: Query to ask what the current subscription list is for a given device. List may be broken up into multiple messages if the list cannot fit within the context of 1 message.
    2. Command: 0xF3
    3. Data sent with command (beyond source address and command): NA
    4. Response: Similar to feedback on the write. Return value includes a count of and a list of how many ERDs are on subscription list.
       1. N = 8- bit qty that represents the count
       2. N – 16-bit ERD #s that confirm the ERD was written.
    5. Example Command:
       1. [0x80, 0x F3]
    6. Example Response: list showing we are subscribed only to model number
       1. [0x80, 0xF3, 0x01 0x00,0x010]
28. ERD Unsubscribe
    1. Purpose: Remove 1 or more ERDs from the subscription list.
    2. Command: 0xF4
    3. Data sent with command (beyond source address and command): Number of ERDs and their identifiers that are to be removed from the subscription list.
       1. N = Number of ERDs to remove
       2. N- 16-bit ERD identifiers.
    4. Response: Simple acknowledgment that reflects the command only.
    5. Example Command: Example is a request to remove Model number form the subscribe list. Note model number should not change so it does not really make sense to subscribe.
       1. [0x80, 0x F4,0x01 0x00,0x01]
    6. Example Response:
       1. [0x80, 0xF4]
29. Subscribed ERD Update Notification
    1. Purpose: Virtually identical to the ERD Write command (0xF1), except the response is a simple confirmation packet. This is published by a node when subscribed ERDs have changed or the subscription periodic has expired.
    2. Command: 0xF5
    3. Data sent with command (beyond source address and command): see 0xF1 description
    4. Response: Packet level acknowledgement.
    5. Example Command: See 0xF1 example but substitute 0xF5 command.
    6. Example Response:
       1. [0x80, 0xF5]
30. Read Data Flash Record
    1. Purpose: Generally used to read the Calibration record results for the upper (0x02) and or lower (0x03) ovens.
    2. Command: 0x95
    3. Data sent with command (beyond source address and command): Flash Record index for the record being requested. Where:
       1. 0 = FCODE
       2. 1 = Dynamic Data (data stored by unit to be restored after reset… time in use etc..)
       3. 2 = Upper oven calibration
       4. 3 = Lower oven calibration
       5. 4 = Special Features
       6. 5 = FCT results
       7. 6 = Fault code
       8. 7 = Periodic Data
       9. 8 = Fault Snapshot
       10. 9 = Cycle History
    4. Response: Record ID and record data that was read returned from the Data Flash. If the record number sent is invalid then an error code is returned in place of the record data.
       1. 1 Byte - Flash Record ID
       2. Then Either:
          1. Record Data:
             1. N *bytes determined by Record type*
       3. *OR*
          1. 0x0E 🡪 indicating bad record type.
    5. Example Command: Example request to read lower oven calibration record.
       1. [0x80, 0x 95,0x03]
    6. Example Response:
       1. [0x80, 0x95, 0x03, …]
31. Calibration Valid Query
    1. Purpose: Used as part of controls testing to ask whether or not the calibration process for a control has been completed.
    2. Command: 0x93
    3. Data sent with command (beyond source address and command): NA
    4. Response: Singe byte returned indicating that the calibration is complete or not. (Complete/incomplete – 1/0)
    5. Example Command: Example is a request for the Model number ERD.
       1. [0x80, 0x93]
    6. Example Response: Response indicating the calibration has been completed.
       1. [0x80, 0x93, 0x01]
32. Read QF Usage Statistics
    1. Purpose: Debug purposes only. Used to read some general loading information related to the architecture framework.
    2. Command: 0x97
    3. Data sent with command (beyond source address and command): NA
    4. Response: 30 bytes of data are returned defining QF usage.
    5. Example Command:
       1. [0x80, 0x97]
    6. Example Response:
       1. [0x80, 0x97, 30 bytes…]
33. Read Boot Loader Parametric Data Item:
    1. Purpose: Command issued to read one of the available boot loader parametric data fields. These fields are read only fields that are populated during the production process.
    2. Command: 0xDD, 0x
    3. Data sent with command (beyond source address and command): a
    4. Response: NA
    5. Example Command: Example is a request for the Model number ERD.
       1. [0x80, 0x F1,0x01 0x00,0x01
    6. Example Response: NA
       1. [0x80, 0xF1, 0x01 0x00,0x010, 0x20,…]
34. Read Boot-loader Parametric Data Item (2-Byte command)
    1. Purpose: Read one of the allocated write once read many locations maintained by the boot loader.
    2. Command: 0xDD 0x03
    3. Data sent with command (beyond source address and command): Parametric Index of value requested. The parametric data corresponding to the parametric index varies by model but the following indexes are fixed:
       1. 1 = Board SKU
       2. 2 = Board Serial Number
       3. 3 = Unit SKU (Model Number)
       4. 4 = Unit Serial Number
    4. Response: Data that represents the parameter or an indication that the data offset requested was invalid. Note that when requesting 0 it is a special case where the return value is an indication of the number of parametric items that exist and the size.
       1. If 0 is requested…then data after command is
          1. N – Byte = number of items that are maintained
          2. S – byte – size of the items (same for all items)
       2. If non-zero valid offset is requested
          1. X – byte – index of parametric item requested (1 – N)
          2. S bytes – data stored at parametric item
    5. Example Command: Example is a request for the Model number ERD.
       1. [0x80, 0xDD,0x03,0x01]
    6. Example Response:
       1. [0x80, 0xDD, 0x03 0x01, Data…]
35. Jump to Boot Loader
    1. Purpose: Forces Application to jump to the Boot loader application for possible software update. Command will only be acted upon if the unit is in a consumer standby (no cooking) or test mode.
    2. Command: 0xDD, 0x07
    3. Data sent with command (beyond source address and command): NA
    4. Response: NA
    5. Example Command:
       1. [0x80, 0xDD,0x07]
    6. Example Response: NA
36. Software Image Valid
    1. Purpose: Allows user to query the validity of one of the software images stored within the control.
    2. Command: 0xDD 0x08
    3. Data sent with command (beyond source address and command): Image ID for the header requested where:
       1. 1 = Boot loader
       2. 2 = Main Application
       3. 3 = Parametric Data
       4. 4 = Auxiliary Image
    4. Response: A structure representing whether the image id was in range and the header date or just that the id was out of range.
       1. Either Case includes:
          1. Byte -- Image ID
          2. Byte – Valid Range (1/0)
          3. U16 – 0x0000 (intended for CRC of header but currently always 0x0000)
       2. If invalid (Valid Range = 1) the remainder of the data is the header for the image – but no more than the max packet that can be transmitted.
    5. Example Command: Example is a request for the Main Application header
       1. [0x80, 0xDD, 0x08, 0x02]
    6. Example Response:
       1. [0x80, 0xDD, 0x08, 0x03,0x01, 0x00,0x00,header data…]
37. Read Memory
    1. Purpose: Used for debug purposes to read data stored at a given address in memory.
    2. Command: 0xDD 0x0C
    3. Data sent with command (beyond source address and command): U8 # of bytes to read (limited by packet size) and U32 start address of read request. If the # of bytes requested exceeds 16 then a 0xFE is returned.
    4. Response: Data read.
    5. Example Command: Read 16 bytes starting at address 0
       1. [0x80, 0xDD, 0x0C, 0x10, 0x00, 0x00, 0x00, 0x00]
    6. Example Response: 16 bytes of data read at address 0
       1. [0x80, 0xDD, 0x0C, 0x10, 0x00, 0x00, 0x00, 0x00, 16 bytes of data]

A second aspect of the command set described above is the individual Entity Reference Designators (ERDs) that are supported within a given control. Each of these ERDs may maintain the ability to be read, written, or subscribed to so an external application can be alerted if the data value changes. See:

* Read = ERD Query 0xF0
* Write = ERD Write 0xF1
* Subscribe = ERD Subscribe 0xF2

Each ERD is characterized by its ERD ID, the length of data associated with the ERD, and an attribute that declares whether the ERD is Read Only. Within the appliance control, ERDs can be useful to control or monitor behaviors of the appliance in the consumer mode.

The following ERDs are supported in the Range (TO12) appliance control:

1. Name: Model Number – (same information 0xDD, 0x03, 0x03)
   1. Designator: 0x0001
   2. Data Size: 0x20
      1. Text characters….
   3. Read Only: True
2. Name: Unit Serial Number (same information as 0xDD, 0x03, 0x04)
   1. Designator: 0x0002
   2. Data Size: 0x20
      1. Text characters
   3. Read Only: True
3. ERD Name: Control UI Lock (front panel lockout)
   1. Designator: 0x0004
   2. Data Size: 0x01
      1. Flag – True/False -- UI Lock/Unlock
   3. Read Only: False
4. ERD Name: Clock Time
   1. Designator: 0x0005
   2. Data Size: 0x03
      1. Hours, Minutes, Seconds (must be valid for each – hours are limited according to user setting.. am or pm)
   3. Read Only: False
5. ERD Name: Clock Format
   1. Designator: 0x0006
   2. Data Size: 0x01
      1. 24/12 Hour Mode – 1/0
   3. Read Only: False
6. ERD Name: Update Display Units
   1. Designator: 0x0007
   2. Data Size: 0x01
      1. Celsius/Fahrenheit – 1/0
   3. Read Only: False
7. ERD Name: Appliance Type
   1. Designator: 0x0008
   2. Data Size: 0x01
      1. Hard coded values indicating type of appliance.
         1. Range = 7
   3. Read Only: True
8. ERD Name: Sabbath Mode
   1. Designator: 0x0009
   2. Data Size: 0x01
      1. Active/Inactive – 1/0
   3. Read Only: False
9. ERD Name: Update Sound Level
   1. Designator: 0x000A
   2. Data Size: 0x01
      1. Level – Off – High – (0 – 3)
   3. Read Only: False
10. ERD Name: 12 Hour Shut Off Setting
    1. Designator: 0x5000
    2. Data Size: 0x01
       1. Enable/Disable – 1/0

Read Only: False

1. ERD Name: Set End Tone
   1. Designator: 0x5001
   2. Data Size: 0x01
      1. Continuous/Beep -1/0
   3. Read Only: False
2. ERD Name: Enable Light Bar (most models do not support)
   1. Designator: 0x5002
   2. Data Size: 0x01
      1. Enable/Disable – 1/0
   3. Read Only: True
3. ERD Name: Convection Conversion
   1. Designator: 0x5003
   2. Data Size: 0x01
      1. Enable/Disable – 1/0
   3. Read Only: False
4. ERD Name: Elapsed Cook Time (not populated in many models)
   1. Designator: 0x5004
   2. Data Size: 0x04
   3. Read Only: True
5. ERD Name: Read F-Code Status
   1. Designator: 0x5005
   2. Data Size: 0x0A
   3. Read Only: True
6. ERD Name: Read Key ID Pressed (coded key id indicates which, if any, key is pressed)
   1. Designator: 0x5006
   2. Data Size: 0x01
   3. Read Only: True
7. ERD Name: Read Oven Configuration
   1. Designator: 0x5007
   2. Data Size: 0x02
   3. Read Only: True
8. ERD Name: Oven Max Temp
   1. Designator: 0x5008
   2. Data Size: 0x04
      1. Current Cook Mode – U16 max temp,U16 min temp in Degrees F

Read Only: True

1. ERD Name: Update Warming Drawer State (not currently implemented)
   1. Designator: 0x5009
   2. Data Size: 0x01
   3. Read Only: False
2. ERD Name: Set Upper Oven Cook Mode.
   1. Designator: 0x5100
   2. Data Size: 0x0D
      1. Data Structure looks like:
         1. U8 u8OvenCookMode (described in data for 5101 below);
         2. U16 u16ExitTemperature;
         3. U16 u16ExitTime; //Cook time
         4. U16 u16ProbeTemp;
         5. U16 u16DelayTime;
         6. U16 u16TwoTempCookTemp;
         7. U16 u16TwoTempCookTime;
   3. Read Only: False
3. ERD Name: Set Lower Oven Cook Mode (same data as Upper oven)
   1. Designator: 0x5200
   2. Data Size: 0x0D
   3. Read Only: False
4. ERD Name: Read Upper Oven State
   1. Designator: 0x5101
   2. Data Size: 0x01
      1. Coded Byte value - Enumerated fields those of interest include:
         1. 0 = No Mode (off)
         2. 1 - 4 = Preheat
            1. 1 = Preheat
            2. 2 = Convection Bake Preheat
            3. 3 = Convection Multi-Bake Preheat
            4. 4 = Convection Roast Bake Preheat
         3. 5 – 10 = Bake
            1. 5 = Bake
            2. 6 = Bake 2 Temp
            3. 7 = Convection Bake
            4. 8 = Convection Bake 2 Temp
            5. 9 = Convection Multi Bake
            6. 10 = Convection Multi-Bake 2 Temp
         4. 11/12 – Convection Roast
         5. 13/14 = Broil Low/Hi
         6. 15/16 – Convection Low/Hi Broil
         7. 17 – Convection Crisp Broil
         8. 18 = Warm
         9. 19 = Proofing
         10. 21/22 = Clean Stage 1/2
         11. 23 = Clean Cool Down
         12. 24 = custom Clean Stage 2
         13. 25 = Steam Clean Stage 1
         14. 26 = Steam Clean Cool Down

Read Only: True

1. ERD Name: Read Lower Oven State (see upper oven state)
   1. Designator: 0x5201
   2. Data Size: 0x01
   3. Read Only: True
2. ERD Name: Upper Oven Delay Time Remaining
   1. Designator: 0x5102
   2. Data Size: 0x02
      1. U16 minutes
   3. Read Only: True
3. ERD Name: Lower Oven Delay Time Remaining
   1. Designator: 0x5202
   2. Data Size: 0x02
   3. Read Only: True
4. ERD Name: Upper Oven Probe Temp
   1. Designator: 0x5103
   2. Data Size: 0x02

Read Only: True

1. ERD Name: Lower Oven Probe Temp
   1. Designator: 0x5203
   2. Data Size: 0x02
   3. Read Only: True
2. ERD Name: Upper Oven Cook Time Remaining (minutes)
   1. Designator: 0x5104
   2. Data Size: 0x02
   3. Read Only: True
3. ERD Name: Lower Oven Cook Time Remaining (minutes)
   1. Designator: 0x5204
   2. Data Size: 0x02
   3. Read Only: True
4. ERD Name: Update Upper Oven Kitchen Timer (Minutes) – starts a countdown timer…
   1. Designator: 0x 5105
   2. Data Size: 0x02

Read Only: False

1. ERD Name: Update Lower Oven Kitchen Timer (not currently supported)
   1. Designator: 0x5205
   2. Data Size: 0x02
   3. Read Only: False
2. ERD Name: Set Upper Oven Temperature Offset
   1. Designator: 0x5106
   2. Data Size: 0x01
      1. I8 - Degrees F
   3. Read Only: False
3. ERD Name: Set Lower Oven Temperature Offset
   1. Designator: 0x5206
   2. Data Size: 0x01
      1. I8 Degrees F
   3. Read Only: False
4. ERD Name: Upper Oven Probe Present
   1. Designator: 0x5107
   2. Data Size: 0x01
      1. Present(connected)/Absent -1/0
   3. Read Only: True
5. ERD Name: Lower Oven Probe Present
   1. Designator: 0x5207
   2. Data Size: 0x01
      1. Present(connected)/Absent -1/0
   3. Read Only: True
6. ERD Name: Upper Oven Elapsed Cook Time (minutes)
   1. Designator: 0x5108
   2. Data Size: 0x02
   3. Read Only: True
7. ERD Name: Lower Oven Elapsed Cook Time (minutes)
   1. Designator: 0x5208
   2. Data Size: 0x02
   3. Read Only: True
8. ERD Name: Upper Oven Display Temperature (Degrees F)
   1. Designator: 0x5109
   2. Data Size: 0x02
   3. Read Only: True
9. ERD Name: Lower Oven Display Temperature (Degrees F)
   1. Designator: 0x5209
   2. Data Size: 0x02
   3. Read Only: True
10. ERD Name: Upper Oven Remote Enable
    1. Designator: 0x510A
    2. Data Size: 0x01
       1. Enabled/Disabled – 1/0
    3. Read Only: True
11. ERD Name: Lower Oven Remote Enable
    1. Designator: 0x520A
    2. Data Size: 0x01
    3. Read Only: True
12. ERD Name: upper Oven Available Cook Modes
    1. Designator: 0x510B
    2. Data Size: 0x0A
       1. Coded modes available -- bit encoded as follows:
       2. Byte 1
          1. [x] Bit 0 = Bake No Option
          2. [x] Bit 1 = Bake Probe
          3. [x] Bit 2 = Bake Delay Start
          4. [x] Bit 3 = Bake Cook Time
          5. [x] Bit 4 = Bake With Timed Warm
          6. [x] Bit 5 = Bake Two Temp Cook Time
          7. [x] Bit 6 = Bake Probe Delay Start
          8. [x] Bit 7 = Bake Cook Time/Delay Start
       3. Byte 2
          1. [x] Bit 0 = Bake CookTime/Warm/DelayStart
          2. [x] Bit 1 = Bake CookTime/TwoTemp/DelayStart
          3. [x] Bit 2 = Bake Sabbath
          4. [x] Bit 3 = Broil Hi
          5. [x] Bit 4 = Broil Lo
          6. [x] Bit 5 = Proof No Option
          7. [x] Bit 6 = Warm No Option
          8. [x] Bit 7 = Warm Probe
       4. Byte 3
          1. [x] Bit 0 = ConvBake\_NoOption
          2. [x] Bit 1 = ConvBake\_Probe
          3. [x] Bit 2 = ConvBake\_DelayStart
          4. [x] Bit 3 = ConvBakeCookTime
          5. [x] Bit 4 = Conv Bake With Timed Warm
          6. [x] Bit 5 = Conv Bake Two Temp Cook Time
          7. [x] Bit 6 = Conv Bake Probe Delay Start
          8. [x] Bit 7 = Conv Bake Cook Time/Delay Start
       5. Byte 4
          1. [x] Bit 0 = ConvBake CookTime/Warm/DelayStart
          2. [x] Bit 1 = Conv Bake CookTime/TwoTemp/DelayStart
          3. [x] Bit 2 = Bake Sabbath
          4. [x] Bit 3 = Broil Hi
          5. [x] Bit 4 = Broil Lo
          6. [x] Bit 5 = Proof No Option
          7. [x] Bit 6 = Warm No Option
          8. [x] Bit 7 = Warm Probe
       6. Byte 5
          1. [x] Bit 0 = ConvMultiBake No Option
          2. [x] Bit 1 = ConvMulti Bake ConvMultiProbe
          3. [x] Bit 2 = Bake Delay Start
          4. [x] Bit 3 = ConvMulti Bake Cook Time
          5. [x] Bit 4 = ConvMulti Bake With Timed Warm
          6. [x] Bit 5 = ConvMulti Bake Two Temp Cook Time
          7. [x] Bit 6 = ConvMultiBake Probe Delay Start
          8. [x] Bit 7 = ConvMulti Bake Cook Time/Delay Start
       7. Byte 6
          1. [x] Bit 0 = ConvMultiBake CookTime/Warm/DelayStart
          2. [x] Bit 1 = ConvMulti Bake CookTime/TwoTemp/DelayStart
          3. [x] Bit 2 = ConvRoastBake No Option
          4. [x] Bit 3 = ConvRoast Bake Probe
          5. [x] Bit 4 = ConvRoastBake Delay Start
          6. [x] Bit 5 = ConvRoastBake Cook Time
          7. [x] Bit 6 = ConvRoastBake With Timed Warm
          8. [x] Bit 7= ConvRoastBake Two Temp Cook Time
       8. Byte 7
          1. [x] Bit 0 = ConvRoast Bake Probe Delay Star
          2. [x] Bit 1 = ConvRoastBake Cook Time/Delay Start
          3. [x] Bit 2 = ConvBroil Low
          4. [x] Bit 3 = ConvBroil High
          5. [x] Bit 4 = ConvBroil Crisp
          6. [x] Bit 5 = ConvBroil Crisp Probe
          7. [x] Bit 6 = Self Clean
          8. [x] Bit 7 = Steam Clean
       9. Byte 8
          1. [x] Bit 0 = Dual Broil
          2. [x] Bit 1 – 7 = NA
    3. Read Only: True
13. ERD Name: Lower Oven Available Cook Modes
    1. Designator: 0x520B
    2. Data Size: 0x0A
    3. Read Only: True
14. ERD Name: WiFi Connected State
    1. Designator: 0x6003
    2. Data Size: 0x01
       1. Connect/Disconnect – 1/0
    3. Read Only: False