# Advanced Database Topics

Bob Dalesio 4-28-99

## Outline

- **Record processing**
- **Database Record Types**
- **Field Access**
- **Conclusions**

# Processing Records - Periodic Scans

- **Periodic Scanning - .1, .2, .5, 1, 2, 5, and 10 second can be modified by editing the . dbd file and adding/deleting the periods desired:**

```
menu(menuScan) {
     choice(menuScanPassive,"Passive")
     choice(menuScanEvent,"Event")
    choice(menuScanI_O_Intr,"I/O Intr")
    choice(menuScan10_second,"10 second")
    choice(menuScan5_second,"5 second")
    choice(menuScan2_second,"2 second")
    choice(menuScan1_second,"1 second")
    choice(menuScan_5_second,".5 second")
    choice(menuScan_2_second,".2 second")
    choice(menuScan_1_second,".1 second")
    choice(menuScan_015_second,".015 second")
}
```

- **Rate is limited by the vxWorks clock tick**
- **Phase orders record processing within the same scan period and priority only!**

## Processing Records - I/O Event

- **Drivers call scanIoRequest to cause records with an I/O Event - SCAN to process:**
  - **in an interrupt service routine**
  - **in an independent thread that reads values, compares to old values and decides a change has occurred**
  - **at any condition that the writer of the driver decides is an event requiring scanning**
- **To initialize event scanning the driver must allocate and initialize some storage for the event:**

```
pFPscanpvt = (IOSCANPVT *)calloc(IP_num_cards*GS_IP_PER_CARD, sizeof(*pFPscanpvt));
if(!pFPscanpvt) return ERROR;
scanIoInit(&pFPscanpvt[card]);
```

- **A routine to return the scan private field must also be supplied:**

```
int fp_getioscanpvt(
unsigned short  card,
IOSCANPVT               *scanpvt)
{
        if(pc_ioc) return(0);
        if (card < (IP_num_cards * GS_IP_PER_CARD))
                *scanpvt = pFPscanpvt[card];
       return(0);
}
```

- **Phase affects the order in which the records are placed in the event queue**

## Record Processing - I/O Event

- **Device support must provide the entry point to get the scan private data from the driver**

```
struct {
    long          number;
    DEVSUPFUN     report;
    DEVSUPFUN     init;
    DEVSUPFUN     init_record;
    DEVSUPFUN     get_ioint_info;
    DEVSUPFUN     read_bi;
} devBiFp={
    5,
    NULL,
    NULL,
    init_bi_record,
    get_ioint_info,
    read_bi};

static long get_ioint_info(
int cmd,
struct biRecord *pbi,
IOSCANPVT *ppvt)
{
  fp_getioscanpvt(pbi->inp.value.vmeio.card,ppvt);
  return(0);
}
```

# Record Processing - Event

- **Events are produced in software by calling *post_event(number)* and cause records to process that have a SCAN of Event with an EVNT of (number)**
- **This functionality is built into the timing and event records. They will generate events whenever the records are processed. This is a method of causing records without event support in the driver to be scanned on event.**
- **State notation programs or any other task in the same vxWorks environment (IOC) can also call post_event().**

# Record Processing - Passive

- **FLNK fields will cause records with SCAN = Passive that have PACT = 0 to process**

- **Any INP field with an attribute of PP (Process Passive) will cause the record specified in the INP field to process if the SCAN = Passive and the PACT = 0**

- **A record containing an INP field with an attribute of CPP will be processed any time referenced record posts a monitor and the SCAN field is also Passive and the PACT = 0. If the INP field has an attribute of CP, the record containing the link processes any time the record referenced in the link posts a monitor regardless of the SCAN field of the record containing the link.**

- **A Channel access put from any channel access client program will cause process a record that has a SCAN of Passive. If the record is currently active, the lock set is locked and the channel access server will wait until completion of the record processing to place the new value into the record. The field being written must have a Process the Record attribute of true - for this to be the case. These are defined in the .dbd file for each record.**
- **Setting the PINI field of a Passive record to True will cause the record to process once on initialization.**

- **Putting a 1 to the .PROC field of a Passive record will cause processing**

## Record Processing - Disable Scanning

- **Conditional scanning can be accomplished using the fields in each database record for disabling the scan.**

- **When the DISV field has the same value as DISA, then the record and any records processed as a result of this record (FLNKs, INPs and OUTs).**

- **SDIS is an address field that specifies the channel from which to fetch the DISA value. If this field is not specified, then whatever is placed into DISA will be used to determine if scanning is disabled.**

- **DISS is the alarm severity of the record having it's scanned disabled. Default is No Alarm.**

## Record Processing - Lock Sets

- A lock set is a set of database records whose FLNKs, INPs, and OUTs, reference each other.
- Lock sets are used to make sets of records execute autonomously - blocking out channel access puts into any of the records until all of the records are processed.
- Links to records in other IOCs are channel access links and do NOT belong to the same lock set.
- To avoid creating this tight binding between records when it is not intended, use the INP modifiers CA, CP, and CPP which force the link to be a channel access link, even if the record is in the same IOC.
- If any of the address fields of a record are changed during operation, any related lock sets will be recomputed.

## Database - Simulation Mode

- **A set of fields to support simulation are supplied on all hardware input records.**
- **SIMM = YES makes this record a simulation record.**
- **A link to a database value to put the record into simulation mode is specified in the SIML. A non-zero number puts the record into simulation mode.**
- **SVAL is the engineering units value used when the record is in simulation mode.**
- **SIOL is a database location from which to fetch SVAL when the record is in simulation mode.**
- **SIMS is the alarm severity of the record if it is in simulation mode.**

# Database - Analog Input

- **Used to read an analog transducer.**

- **Conversions for linear transducers accomplished using LINR = Linear and placing the full range of the conversion in EGUF and the low end of the range in EGUL**

- **A weighted average is built in for smoothing in the form**

  **VAL(new) * (1-SMOO) + VAL(previous) * SMOO**

- **Break point tables used for non-linear conversions or tables.**

## Analog Input - Breakpoint Tables

- **To add a break point table enter the breakpoints into the .dbd file with counts in the first column and engineering units in the second column**
-

```
breaktable (IGtable){
    0, 0,
    2048, .00000000001,
    2068, .000000000011,
        :
    4077, .085,
    4097, .1
}
```

- **Add the breakpoint into the conversion file: menuConvert.dbd**

```
menu(menuConvert) {
    choice(menuConvertNO_CONVERSION,"NO CONVERSION")
    choice(menuConvertLINEAR,"LINEAR")
    choice(menuConverttypeKdegF,"typeKdegF")
    choice(menuConverttypeKdegC,"typeKdegC")
    choice(menuConverttypeJdegF,"typeJdegF")
    choice(menuConverttypeJdegC,"typeJdegC")
        :
    choice(menuConverttypeSdegC,"typeSdegC")
    choice(menuConverttypeIG,"IGtable")
}
```

## Database - Analog Output

- **VAL is the engineering units of the value requested as the output.**
- **If OIF is Incremental, then the OVAL is set to be the previous OVAL + VAL.**
- **The VAL is only changed by OROC on every scan as long as OROC is a non-zero number and OIF is Full.**
- **After a new value for OVAL is computed, it is limited by DRVH and DRVL. These are software drive limits on the output.**
- **Also note the fields that are used by all output records for performing closed loop control and reacting to errors on the values used to determine the new outputs.**
- **OVAL is converted to a binary number and sent to the hardware.**

## Database - Binary Output

- **The binary output is a momentary output if the HIGH field is set to a non-zero value. If HIGH is .5 and the VAL is set to 1, .5 seconds later the VAL is set to 0 and the record is processed.**

- **Also note the fields that are used by all output records for performing closed loop control and reacting to errors on the values used to determine the new outputs.**

# Database - Calculation

- **12 inputs that are specified in the fields INPA through INPL**
- **The values are fetched and stored in the CALC record in the fields A though L. These values can be entered directly into the A through L fields if their respective INP<x> is not defined.**
- **Algebraic Operators: ABS, SQR, MIN, MAX, CEIL, FLOOR, LOG, LOGE, EXP, ^, **,+, -,*,/,%, NOT**
- **Trigonometric Operators: SIN, SINH, ASIN, COS, COSH, ACOS, TAN, TANH, ATAN**
- **Relational Operators: >=, >, <=, <, #, =**
- **Logical Operators: &&, ||, ! (Not)**
- **Bit-wise Operators: |, &, OR, AND, XOR, ~, <<, >>**
- **Parentheses and nested parens are supported**
- **The C '?' operator is supported with nested conditionals (expr)?(true):(false)**
- **36 characters are allowed for the expression**
- **The Infix expression is converted to byte code reverse polish for execution**
- **When the expression is changed during operation, a new postfix byte code is generated**

## Database - Calculation Output

- **Like a calculation in all ways except that it includes an OUT field for writing the result.**
- **The OOPT field controls when a value is written and includes the options: every time the record is scanned, when the value changes, when the value is zero, when the value is not zero, when the value transitions to zero, when the value transitions to not zero.**
- **When DOPT is Calc, then the result of the initial expression is used for output. If the DOPT field is set to OCAL, then the result of the OCAL expression is used for output. This allows the CALC to be used to determine if a value should be output and OCAL then to be the value to output.**
- **It the OEVT field is non-zero and the OOPT permits the output, a post_event is called with the event number from the OEVT field.**
- **To delay before the output is sent, set the ODLY field to the second for wait until the output is set. The record has PACT set during this period and further sets DLYA to true while waiting.**
- **Finally, the record features many status values: those of the link status for each INPx field, CALC valid, OCAL valid and the output address valid.**

## Database - Compression Record

- The compression record is used to compress arrays or to keep some historical data from an scalar. It runs as a circular buffer, a successive average or to compress values.

- When ALG is Circular Buffer, it reads the scalar referenced by INP and places the value into the next element of the NSAM element array.

- When ALG is Average, INP is treated as an array (of possibly 1 element). Each time the record is processed the new array is added to the sum. On the Nth read, the array is averaged and placed into the value of this record. (The Record Reference Manual is incorrect on this record)

- When ALG is any of the N to 1 algorithms (High, Low, Average), and the INP is an array, the INP array is read and values are discarded until one of them is lower that the filter value ILIL or higher than the IHIL. If ILIL and IHIL are 0, no checking is done. The array is then compressed according to the algorithm - every N samples becomes one sample in the new array and is either the High, Low or Average of the N.
- If the INP is a scalar, then no filter is used. N successive reads of the scalar are done to produce one value in the resulting array.
- (The record reference manual mentions an OFF field which does not appear in the code)

## Database - CPID (Cebaf PID)

- **A control algorithm that changes an output value based on the error between the user specified setpoint and the readback value. This implementation of the algorithm is:**

- **$\mathrm{delM}(n) = Kp\,(E(n)) - E(n-1)) + Ki * E(i) * dT(n) + Kd * \dfrac{E(n) - E(n-1)}{dT(n)} - \dfrac{E(n) - E(n-1)}{dT(n-1)}$**

  **proportion + integral + derivative**

- **When OMOD is Change then OVAL is set to DM (The analog output would need to have OIF set to Incremental for an absolute analog output - Absolute for a motor controller). If OMOD is POSITION, the OVAL is set to OVAL + DM.**

- **LOC, MMOD and SMOD can change the operation from the above description into an override situation. See the record reference manual.**

- **MIN and MAX limit the OVAL and DMIN and DMAX limit the DM (change in the output).**

## Database - Fanout

- **Used to fanout the processing link FLNK to more than one record. This record contains 6 forward links.**

- **As FLNKs are processed as a recursive subroutine call, using a FANOUT block is a way to limit the use of the stack on very large processing chains. This record is also useful for the conditional processing of sets of records.**

- **SELM sets the mode for executing the links. ALL - processes all links in order. SELECT will process only the link number set in SELN. MASK will process all links up to SELN.**

- **When the address field SELL is defined, a database value is read and placed into SELN.**

## Database - Data Fanout

- **Used to fanout the same value to up to 8 database locations. This may be useful for placing a change of an alarm limit to many records.**

- **Uses the OMSL and DOL fields like all output records.**

## Database - Histogram Record

- **Keeps hit counts of the number of times a value was in a certain value range. This array can be used to monitor range of operation for a given signal and watch for excursions outside of the normal operating range.**
- **SVL specifies the location of the value.**
- **There are NELM buckets of equal size that range between ULIM and LLIM.**
- **This record features a timed monitor parameter, SDEL, that causes a callback to be executed every SDEL seconds to post this monitor.**

## Database - Multi-bit Binary In/Out

- **Define up to 16 states with 32 bit value patterns .**
- **Specify the starting bit number in the INP/OUT field along with the number of bits used (NOBT) that must be adjacent.**
- **Converts a bit pattern into a state.**
- **Non-matching bit patterns have an alarm status of STATE ALARM and a value of 65535.**
- **If no state strings or state values are defined, the bit pattern is right justified and placed in the VAL field.**
- **Typically the RVAL field is the bit pattern read from the hardware and right shifted to have the least significant bit as bit 0. This is the responsibility of the device support - so there is no guarantee.**

## Database - Multi-bit Binary In/Out Direct

- **Reads in up to 16 bits without needing name space and records to support each bit.**
- **Specify the starting bit number in the INP/OUT field along with the number of bits used (NOBT) that must be adjacent.**
- **Individual bits are addressed through fields B0 through BF.**
- **RVAL typically has the right justified value read from the hardware and shifted right by NOBT.**

## Database - Scan Record

- **Used to set up data taking experiments using up to four controllers collecting data from up to four input channels.**

- **This N dimensional control space can be coordinated through a series of records to wait for movement to be complete before collecting data.**

## Database - Select

- **The VAL field is the result of a selection of up to 12 value from INP links.**
- **The value is selected based on the mode set in SELM. SELM of High, Low or Median, selects based on relationship to the other values. An SELM of Specified uses the NVL field to fetch the link to use, or the SELN if there is no link defined for NVL.**

## Database - Sequence Record

- Provides a mechanism for executing up to 10 read and/or write cycles with defined delays.
- Each cycle reads a value through an optional DOLx and then writes the value to an optional LNKx. (Remember: address fields with constants set the value of the D0x field on initialization and subsequent writes will write whatever value is currently in D0x.)
- A configurable delay is available between each cycle.
- When the select mode (SELM) is ALL, each of the 10 cycles is performed. When SELM is SELECT, a value fetched through the SELL field or the SELN is used to determine which of the 10 links to execute.

## Database - Steppermotor Record

- **Runs in either Position or Velocity mode. In Velocity mode. When MODE is Velocity, then the VAL field is the steps per second. In Position mode, the value is the absolute position.**

- **As with all output records, the OMSL is used to determine if the VAL comes from the DOL field (closed-loop mode) or if the VAL field is used directly (supervisory mode).**

- **The value used as the readback for the position is a database link called RDBL. The position feedback can come from an analog input record reading a position transducer or the motor position set in the driver (MPOS) or the encoder position that was readback from the controller (EPOS).**

- **Retries are configured by setting the RTRY field to the number of retries desired until the RBV and the VAL are within the specified deadband (RDBD). The error on the first move is saved in the MISS field. RCNT contains the last number of retries used.**

- **A motor can be configured to initialize to one of the limits by setting the IALG field. The initial value of the motor would be set be the IVAL field. The motor is given the Home command - which causes the current position to be 0. Every position readback is offset by the IVAL when the IALG is set to move to a limit at initialization.**

## Database - Steppermotor Record - continued

- **Setting a 1 in the STOP field will cause a stop command to be sent to the motor.**
- **MPOS is the value read back from the motor controller of the current motor value converted to engineering units. EPOS is the encoder readback (when available) converted to engineering units. Setting the HOME field to 1 sends a command to the motor to set the current encoder and motor positions to 0. These values are available regardless of the RDBL.**
- **Other motor status available includes: MCW and MCCW to indicate the state of the clockwise and counter clockwise limits (only meaningful for analog clocks). A one indicates the limit switch is hit. If both are 1, the motor will not move. The MOVN field indicates that the motor is in motion. The DMOV field is set only after all motor retries are complete. The DIR field gives an indication of the travel direction of the motor.**
- **ERES and MRES are the resolution of the motor and encoder. DIST is the engineering units distance of 1 pulse. Configuring these parameters incorrectly can result in no movement from the motor due to the conversion of the velocity and acceleration going outside of the proper range. When starting with a motor, it is best to keep DIST as 1 so that the VAL is in units of steps. When the motor resolution MRES (motor pulses per revolution) and the encoder resolution ERES (encoder pulses per revolution) are correct, then set up the motor to be in the desired units.**

## Database - Subroutine Record

- Calls a C subroutine in the vxWorks environment in the context of the database execution.
  **The subroutine specified by INAM is called the first time the record is executed and then SNAM is called each subsequent time.**

- **12 INP links are available to read up to 12 values into fields A-L.**

- **BRSV is the alarm severity if the subroutine returns less than 0.**

- **Your subroutine gets a pointer to the database record structure of the calling record. This means that all fields of the subroutine record may be accessed and altered in the context of the subroutine. You can also use this mechanism to reference records with arrays, manipulate the arrays inside the subroutine and place the resulting array in the VAL field of a waveform record referenced by one of the INP links for this subroutine record.**

- **Refer to the record reference manual for examples of asynchronous record completion.**

## Database - Field Modification

- **Most fields of a database record can be modified during operation. That is, they are allowed by channel access and the definition of the field in the .dbd file.**

- **Some of the field changes cause other fields to change and require that the record support have special processing to support the change. EGUL and EGUF are good examples that are broken in 3.13.1. The CALC field is another that requires that the CALC be converted to byte code postfix notation.**

- **All address fields can be modified during operation - but not all device and driver support are written to work with this. Database address changes are supported - locksets are recomputed to reflect the new and broken relationships.**

## Database conclusions

- **The database is provided as a mechanism to implement closed-loop control and data acquisition without programming in a traditional language. However, as the database describes process flow and data flow, it has much of the complexity of a program.**

- **Some records are provided to implement even some sequential control logic and for loop type operation (SCAN and SEQUENCE records). These are fine to use for a limited scope problem.**

- **Database designs can be greatly complicated by not understanding the was each record functions and the features that it supports. This can be mitigated by having group reviews. These go a long way to producing a common style - or at least an understanding of what others are doing.**

- **The set of records supported is extendible. If you have a problem that could be encapsulated into a record type and is used in many instances, consider this as an option.**

- **Record types are increasingly being used as an interface to device and driver support to provide diagnostic information to channel access and allow configuration parameters to be set over channel access.**