

SUNGARD[®] PUBLIC SECTOR

ONESolution[™] 

FINANCE

Budgeting Guide

SUNGARD PUBLIC SECTOR, Inc. NO REPRESENTATIONS OR WARRANTIES, ORAL OR WRITTEN, EXPRESS OR IMPLIED, WITH RESPECT TO THE SYSTEM, SERVICES, SOFTWARE, DOCUMENTATION, OPERATING ENVIRONMENT, ANY OTHER SOFTWARE OR SERVICES PROVIDED HEREUNDER OR ANY OTHER MATTER ADDRESSED HEREUNDER, AND SUNGARD BI-TECH EXPLICITLY DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF TITLE, MERCHANTABILITY AND FITNESS FOR A SPECIFIC PURPOSE. SunGard Public Sector, Inc shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material. This documentation is proprietary and confidential information of SunGard Public Sector, Inc. Copying, reproduction or distribution is strictly prohibited. All rights reserved.

Copyright © 2009 by SunGard Public Sector, Inc.
890 Fortress Street
Chico, CA 95973



DRAFT

1.1 INTRODUCTION

This guide is formatted to be used as you are lead through this training. We will attempt to follow the flow of the user guide as closely as possible, but we will have some flexibility as to the depth we decide to discuss each topic.

Budget Training typically lasts somewhere between one and two days. The material covered will remain consistent regardless of the length of the training. A two-day training schedule will go into more detail and may include more examples than a shorter training schedule.

The training will be separated into the following activities:

- ✓ Account Structure Review
- ✓ Review of Budget Concepts - Budget control and checking
- ✓ Review of and practice with different budget entry methods
- ✓ Budget reporting functionality review

The goal of this training and this training guide is to help you build a solid foundation that will allow you to effectively create, enter, control, and report on budgets within your organization. Though the trainer, the subject, and the training materials can all add to the overall success of a training, the key is your active participation and interest.

This is your training. Any and all questions are encouraged and appreciated.

1.2 GENERAL LEDGER STRUCTURE REVIEW

Before we begin to talk about IFAS Budgeting it is first necessary that you understand your IFAS account structure. To do this it will be necessary for everyone to have a copy of the General Ledger - General Information Report - GLREFLGN.

An understanding of this structure is critical since reports will allow us to pull out information based on all of the background parts that were created during the initial set up of your account structure.

Other useful reports include:

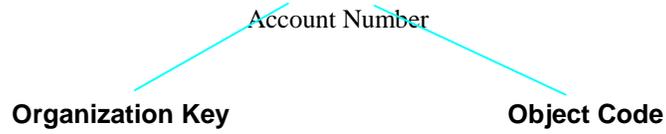
GLREFLOG - Listing of all valid Object Group values

GLREFLPT - Listing of all valid Organization Part values

GLREFLKY* - Listing of all valid Organization Keys

GLREFLOB* - Listing of all valid Object Codes

- - These file listings can be quite lengthy so print them with caution.



Org. Parts	Example Value	Example Description	Object Groups	Example Value	Example Description

1.3 CREATE AND SET UP BUDGET VERSIONS

To begin using the budgeting functions in IFAS, at least one budget version has to be created. IFAS allows a maximum number of 25 budget version.

Any budget version exists in parallel for any fiscal year.

Budget Versions are not limited to hold annual budgets, they can also be used to hold actual data for prior fiscal years.

Budget Versions are set up in GLUPGN – Budget tab.

1.3.1.1.1 GLUPGN Budget tab

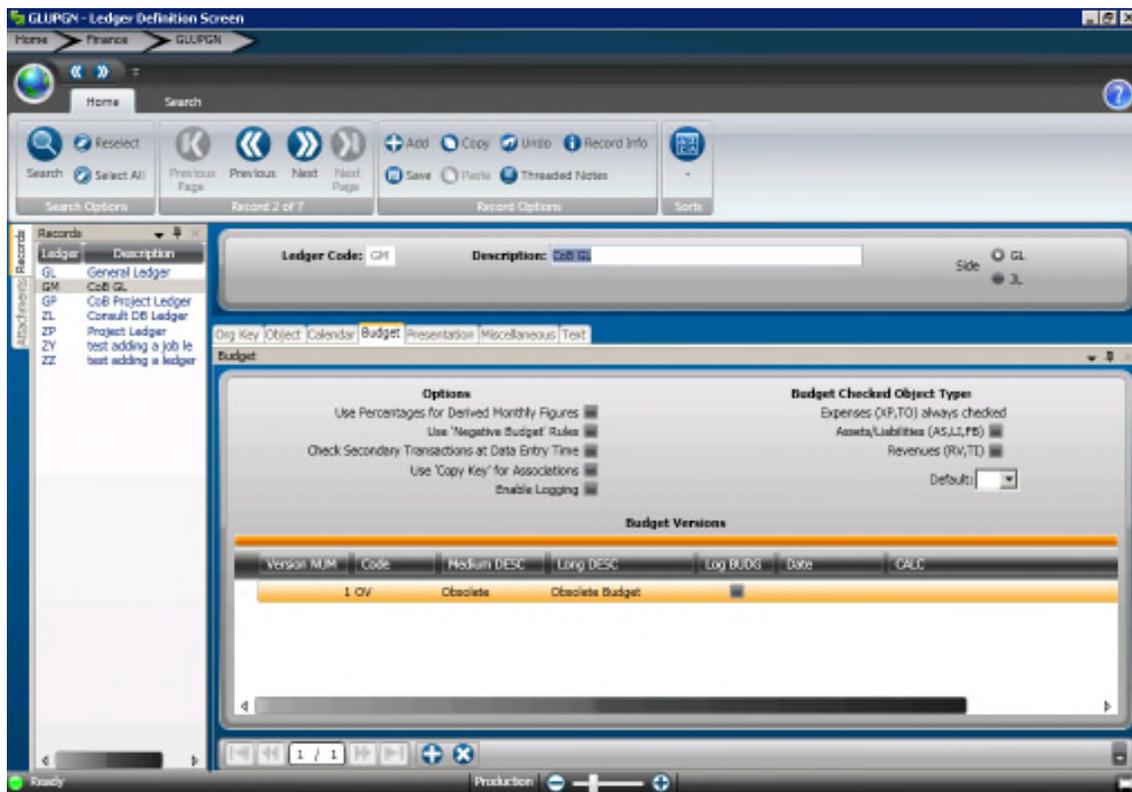


Figure 1: GLUPGN - Budget Tab

The Budget tab within the Ledger Definition Screen defines a ledger’s budget structure.

Enable Logging: Placing a checkmark in this field allows logging for this ledger, and is enabled or disabled on a version by version basis on by placing a checkmark to the right of the budget name definition fields.

Budget Checked Object Types: This area controls which object types will be checked against budgets. Expenses and Transfer Outs are always checked.

Version NUM / Code: A two-character alpha-numeric field that will represent budget within the general ledger transaction and accumulator tables. There are 25 available budget versions.

Medium Desc.: A maximum of eight characters which describe the Budget Name.

Long Desc.: A maximum of 16 characters which describe the Budget Name.

Date: Assign a cutoff date to the budget version if desired.

Calc: The Calc field can be used for a couple of functions:

Track actual figures in a budget version by putting ‘AC’ in the date field. -0001AC will pull in prior years actual.

Define a budget version to be derived from other budget versions. For example, we may want to have our Working Budget be derived from the combined values of our Approved Budget plus the values of our Adjustments Budget. Assuming that the two character versions for the Approved and Adjustments Budgets were AP and AJ, then the Calc Code field would read ‘AP+AJ’.

Log: A checkmark here turns on logging for this budget version.

Default version: The value in this dropdown box defines the default budget which will be used for all budget checking and reporting. This default can be overridden in GLUPKY (org key definition).

1.3.1.1.2 GLUPGN tab Miscellaneous

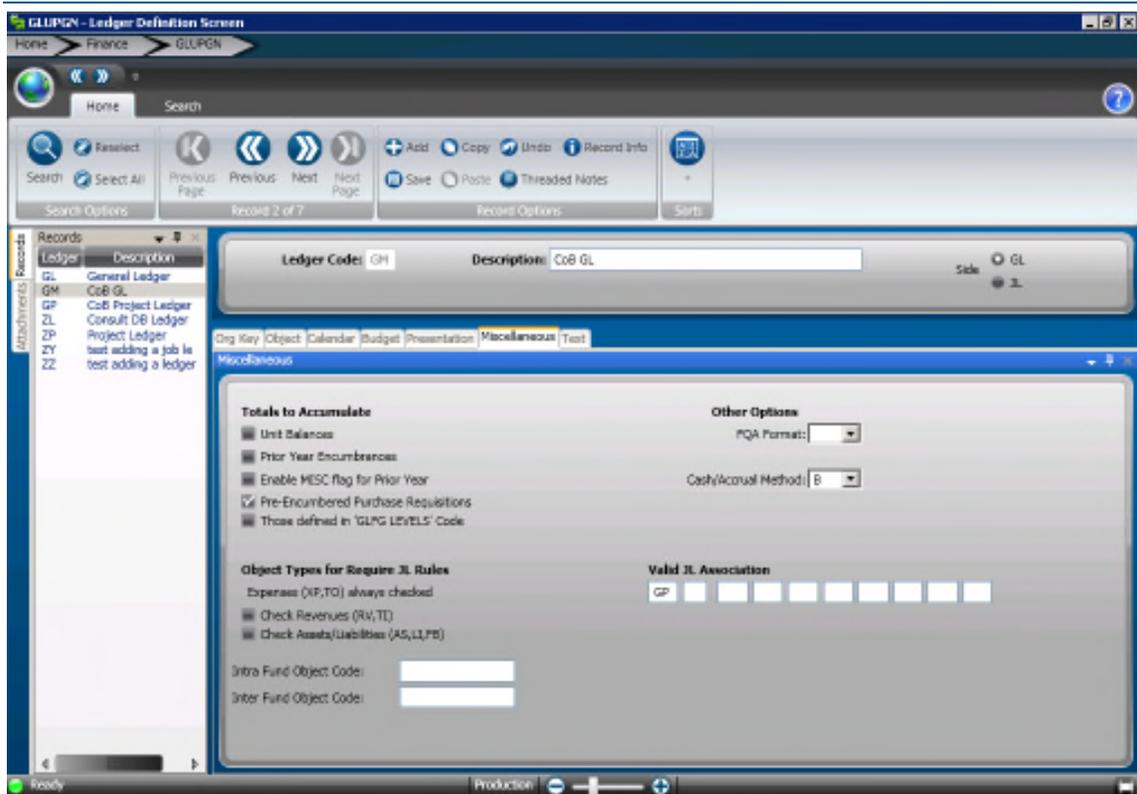


Figure 2: GLUPGN - Miscellaneous Tab

Totals to Accumulate: A checkmark next to ‘Those defined in ‘GLFG - LEVELS’ code’ allows us to use the common code to specify at which object and object group levels to

maintain accumulators. This allows us to set up global budget warning or blocking at any object group level.

1.4 BUDGET CONTROL

1.4.1.1.1 Blocking versus Warning:

When IFAS is set to check the availability of funds for transactions there are two types of budget alerts available:

Blocking will force budget to be available before a batch can be created or distributed.

Warning will cause a message that identifies which account numbers do not have sufficient budget available to process a transaction, but will allow the batch to be created and posted.

There are three different places that blocking and/or warning can be established:

1.4.1.1.2 Global blocking or warning - Common Code - GLFG-LEVELS

It may be desirable to establish budget alerts that are consistent for all keys and objects that are subject to budget checking. This is done through the use of the Common Code GLFG/LEVELS. Any budget blocking or warning set up here will be in effect for all keys and object codes.

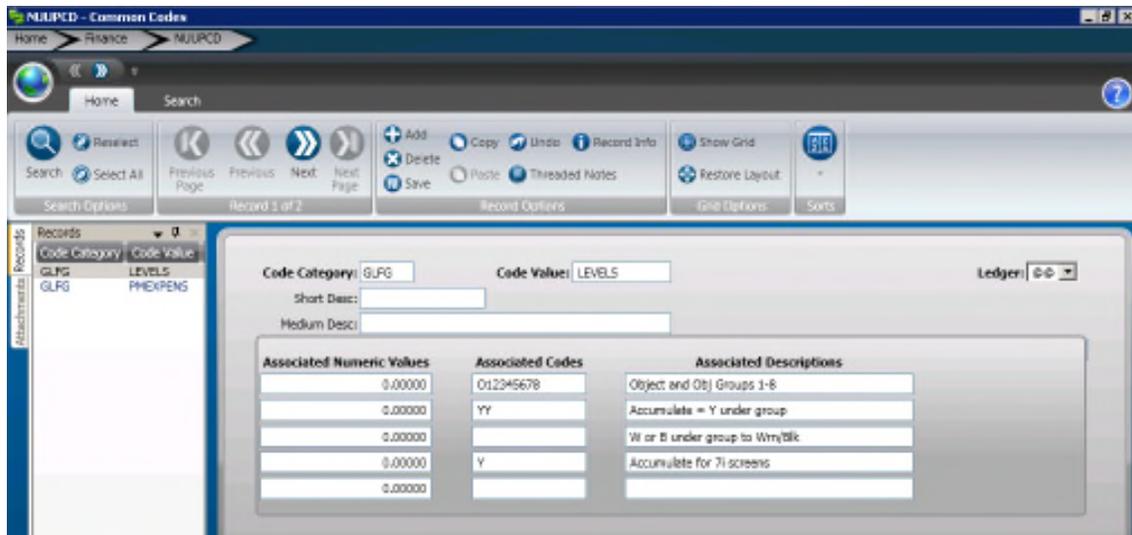


Figure 3: NUUPCD - GLFG / LEVELS Code

First Associated Codes field:

Represents available budget levels (O for Object Code, 1-8 for Object Groups 1-8 as defined on the Object tab of GLUPGN).

Second Associated Codes Field:

A ‘Y’ in this field sets up an accumulator level corresponding to the levels in the first Codes field (Object level, Object Group 3, and Object Group 5 from the example above).

Third Associated codes Field:

The third Codes field sets up blocking with a ‘B’ and/or warning with a ‘W’ for the levels as indicated in the first Codes field.

Fourth Associated Code Field:

Repeat Third associated code field for 7i

1.4.1.1.3 Blocking or Warning by individual Organization key - GLUPKY

In may be necessary to define different budget alert options for different areas of the organization. For example some departments may want to block every time Object Group 3 was exceeded and others may want to block every time that a line item Key-Object combination is over budget. This can be accomplished by setting budget warnings on a key-by-key basis.

The ‘Over Budget Flags’ section is where we set up budget alerts on a key by key basis. The ‘Action’ field sets the type of control: ‘B’ for block or ‘W’ for warn. The ‘Level’ field indicates at what level this is to occur. Our choices are either ‘O’ for Object Code level or G1-G8 for Object Group 1-8. In this example the field has ‘B3’ indicating block at the third Object Group level.

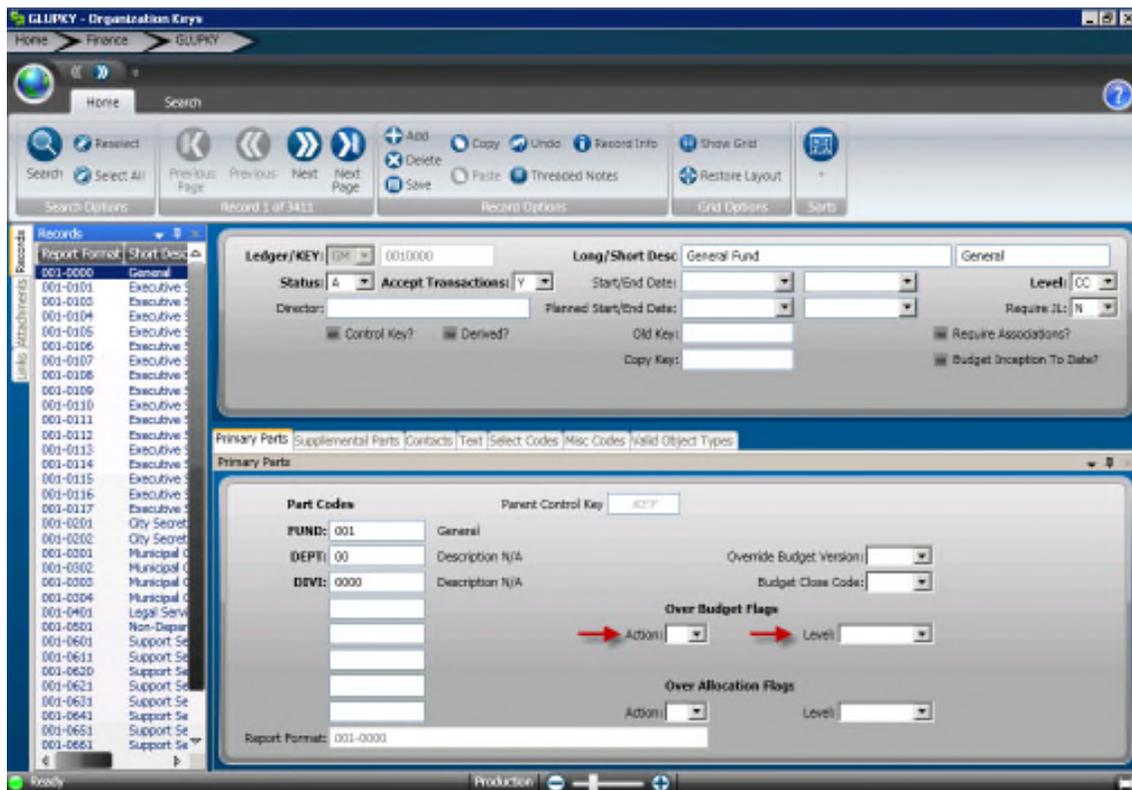


Figure 4: GLUPKY - OVER BUDGET FLAGS

1.4.1.1.4 Blocking or Warning by Individual Account Numbers - GLBUUP

Some organizations may want to set up budget alarms on an account number by account number basis. For example a division may only want to be warned when they are over budget for operating supplies but they want to be blocked if they are going over budget for payroll expense. This level of control is available through the use of the Direct Budget Update Screen.

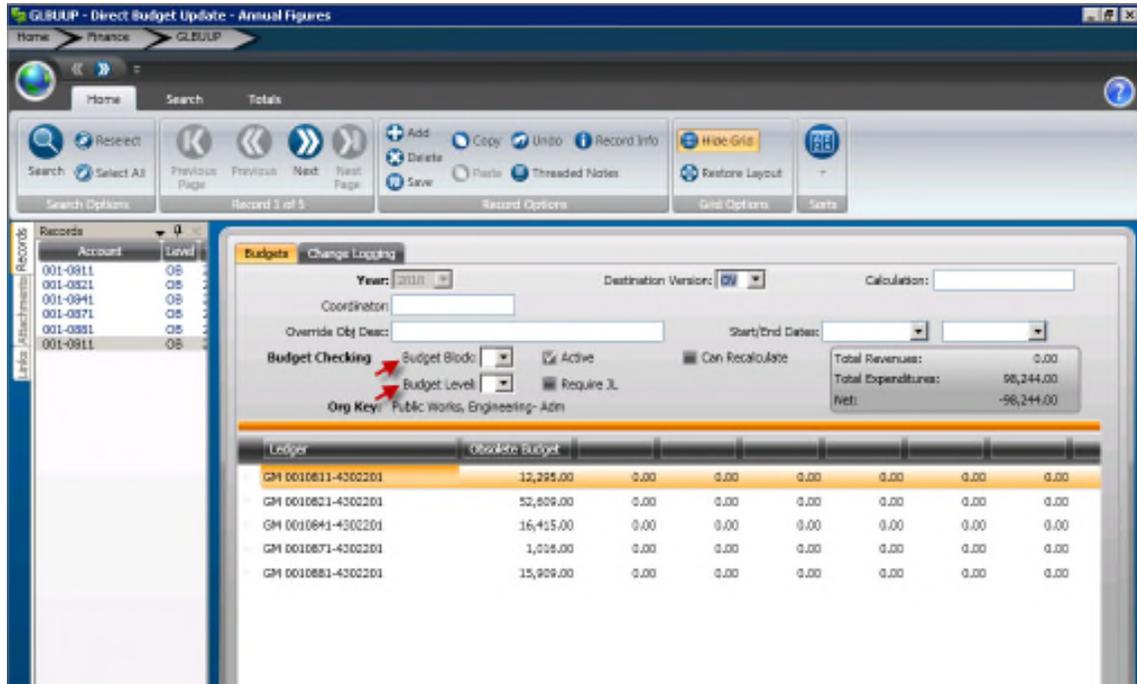


Figure 5: GLBUUP - BUDGET BLOCK

On this screen there are Budget Block and Budget Level fields associated with every line item that works the same way as the ‘Budget’ field on the GLUPKY screen. In our example

1003200-54300 would block at the account level.

*Because there are three different areas that budget alarms are set it is possible to have some account numbers being set up to have multiple budget checking attached to them.

For example: GLFG LEVELS is set to Warn at Object Group 5 and GLUPKY is set to Block at Object Group 3 for key 1003200. Every time a transaction is entered for key 1003200 the user will be blocked if the key is over budget at Object Group 2 and will be warned if it is over budget at Object Group 5. Blocks win.

1.4.1.1.5 Control Keys

We have already discussed the fact that it is possible to budget check based on any of our Object Group levels, but what if we want to pool budgets based on an Organization Key Part?

To facilitate this functionality IFAS allows the creation of Control Keys. This allows us to create keys whose only purpose is to pool budget values for budget checking purposes.

For example: Budgets have been established for all keys in department 1200. For budget checking purposes we do not care whether or not any individual key within the department goes over its budget. We want to stop that department from going over its total budget for any object code. We set this functionality up through the use of Control Keys.

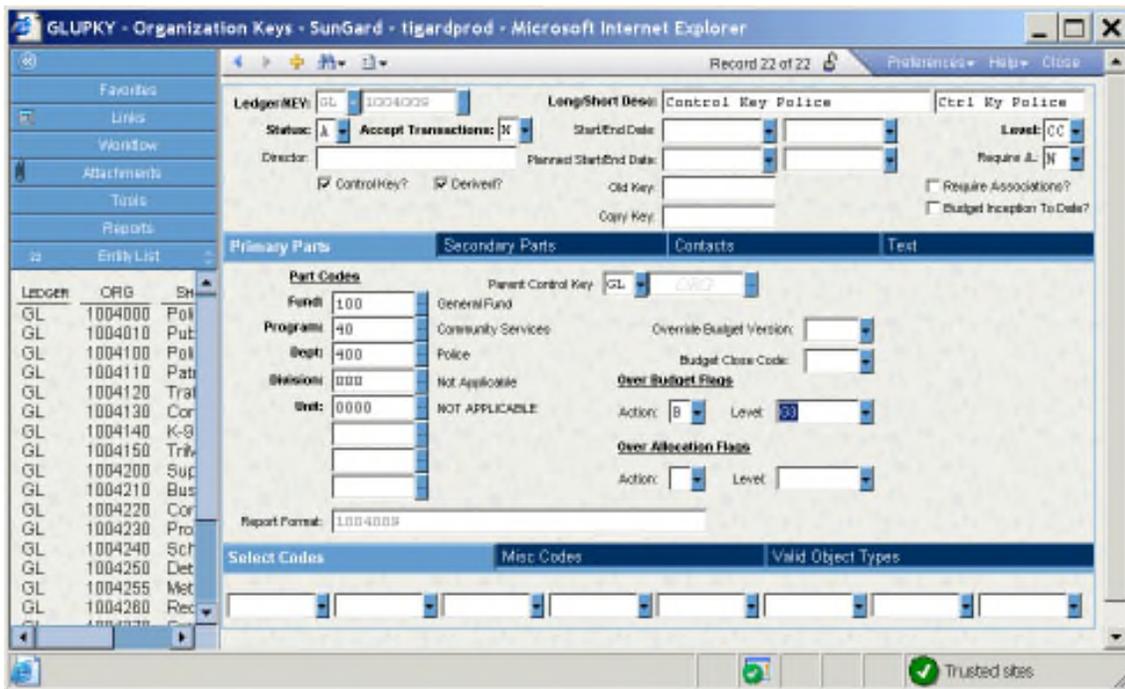


Figure 6: GLUPKY - Control Key Set Up

This is the control key set up to pool budgets for all keys with a department value of 1200. The important fields:

Control Key?: A checkmark indicates that this is a Control Key.

Accept Tr?: This becomes ‘N’ indicating that this key will not accept transactions.

Again the only purpose of this style of control key is to accumulate other keys budgets for budget checking.

Derived?: A checkmark indicates that the budget values for this key will be derived from the budget values entered at the lower level keys.

You will also notice that the ‘Over Budget Flag’ field is set to block at G3, meaning that if the pooled budgeted values for Object Group 3 for the keys pointing to this Parent Control Key for budget checking is exceeded by a transaction from one of the lower level keys that transaction will be blocked.

The following set up is necessary on the key creation record for any lower level key that we want to have point to a Control Key:

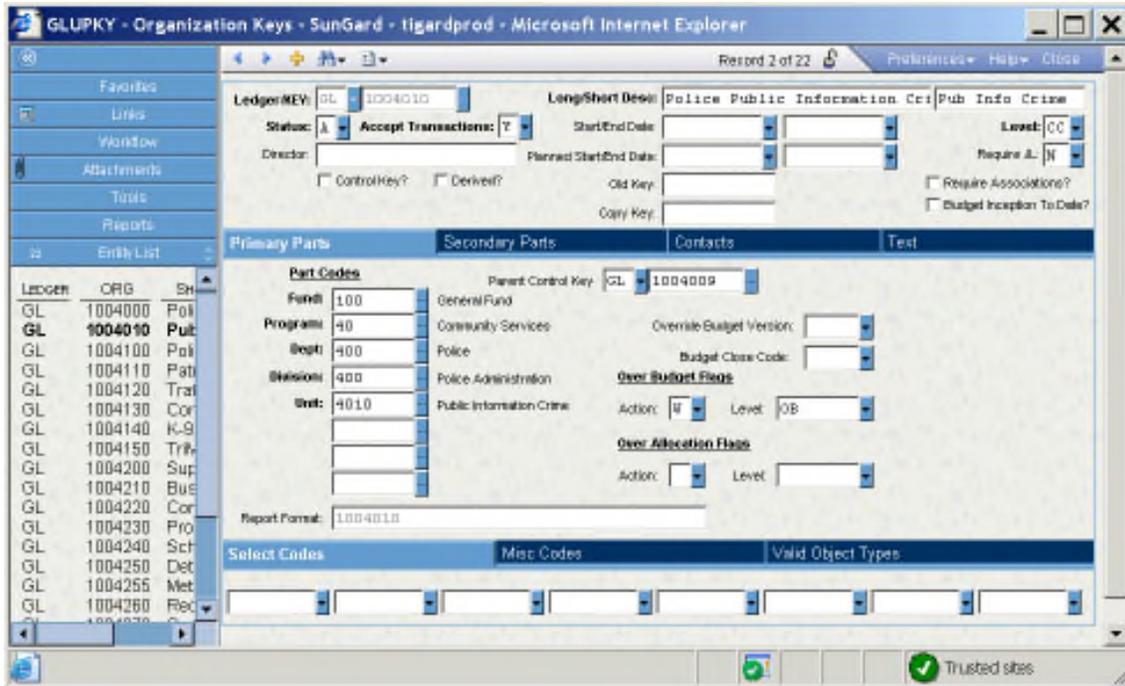


Figure 7: GLUPKY - Child Key pointing to Parent Control Key

The value of the Control Key that we want to use for budget checking for this key needs to be entered in the ‘Parent Control Key’ field. In this example all transactions will have their budget checking done from the pooled budget values accumulated within Control Key 1004009.

Note: when using Control Keys it is important to remember that Control Keys will not show for most Budget to Actual Reporting unless they are specifically asked for. The logic behind this is that since Control Keys are really only a pooling place for lower level keys, values would be double reported if both the lower level keys and Control Keys are shown on a report.

1.5 BUDGET LOGGING

Before going through the various budget data-entry modes available, we need a discussion relating the concept of budget logging. In GLUPGN tab Budget, we enabled budget logging for the system and enabled logging on a version by version basis.

Budget logging will maintain a log of all changes made to a given budget version and then can be reported on through the use of either GLREFLCB or CDD. The log will track IFAS Job, date, user, amount, and budget entry method, reason and funding type. This allows us to easily identify any changes that may have taken place.

1.6 BUDGET ENTRY METHODS

Following methods are available for budget entry within IFAS:

- ✓ Direct Budget Update – GLBUUP
- ✓ Direct Budget Update from Flat File - GLBUFCF
- ✓ Budget Batch Adjustments - GLBUBAUB
- ✓ Intra-fund Budget Methods - GLBUBAAJ, or GLBUBAAN
- ✓ Copy Budgets - GLBUCB
- ✓ Budget Models – GLBUBM
- ✓ Budget item detail - BD

As you will see, each has its purposes, advantages, and disadvantages. The important thing to take from this training is which methods will work best for your organization, and at what stage your organization is likely to utilize each budget entry method.

1.6.1.1.1 Direct Budget Update (GLBUUP)

The most frequently used and most direct method for budget entry into IFAS is Direct Budget Update. Using this method directly updates budget values and does not require any batch processing.

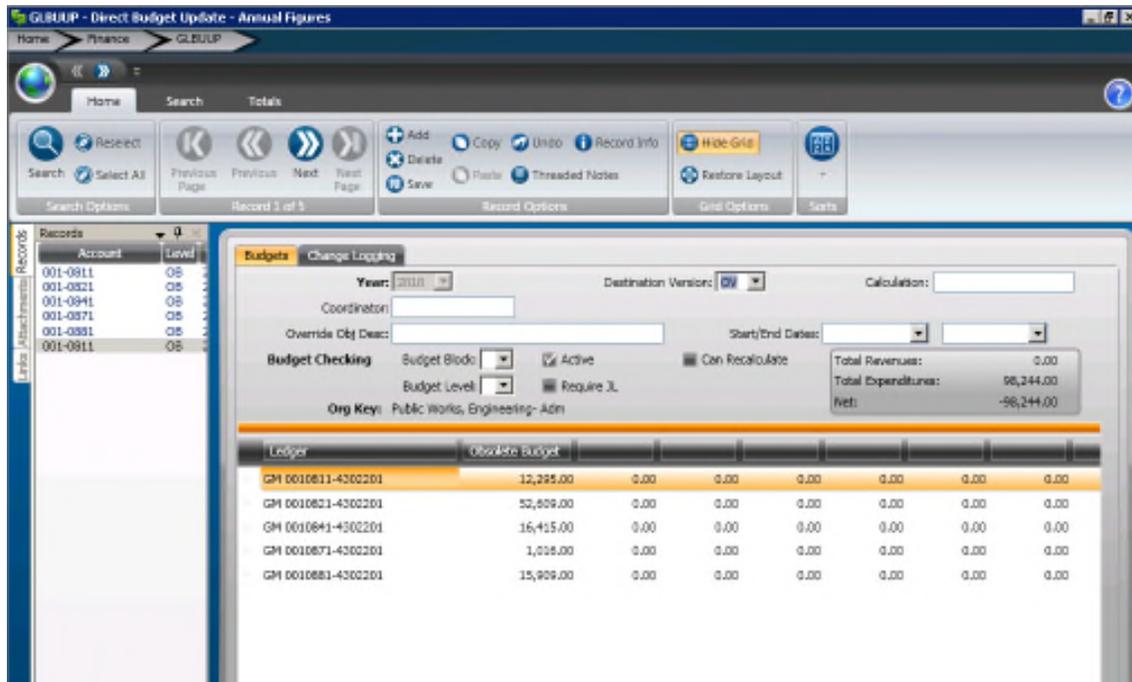


Figure 8: GLBUUP - Direct Budget Update Screen

Though at first glance there appears to be a lot of information required to use this screen, we will see that budget entry is fairly simple and expedient. Attached is the on-line IFAS documentation for this screen accessed by pressing the f1 function key.

1.6.1.1.2 GLBUCF – Load Budgets into Database from Flat File

Often times, especially early in an implementation it is easiest to load the original budget from a flat file either created by a download from the legacy system, run through an account conversion table or from a spreadsheet of some kind.

All that IFAS needs to create an annual budget is a tab delimited text file that has the Organization Key, Object Code and Dollar Amount.

The file is saved as a tab delimited text file and transferred to the server. Once on the server the file is padded to 80 bytes and the program GL942 will convert the file and load it into the system.

1.6.1.1.3 Budget Batch Updates

Though the Direct Budget Update method of budget entry is very convenient and relatively quick, there are times when organizations need to be able to go through a more structured process for budget entry and adjustments. Budget Batch Updates allow a user to create a budget batch, run a batch-proof, get any necessary approvals and then distribute the batch. The changes will not go in to affect until after the batch has been distributed.

All batch updates to budgets are automatically logged regardless of whether logging was enabled on the General Information screen.

The method that allows the most flexibility is GLBUBAUB. It allows budgets to be entered to multiple funds and does not require a balanced entry.

Go to Mask GLBUBAUB01

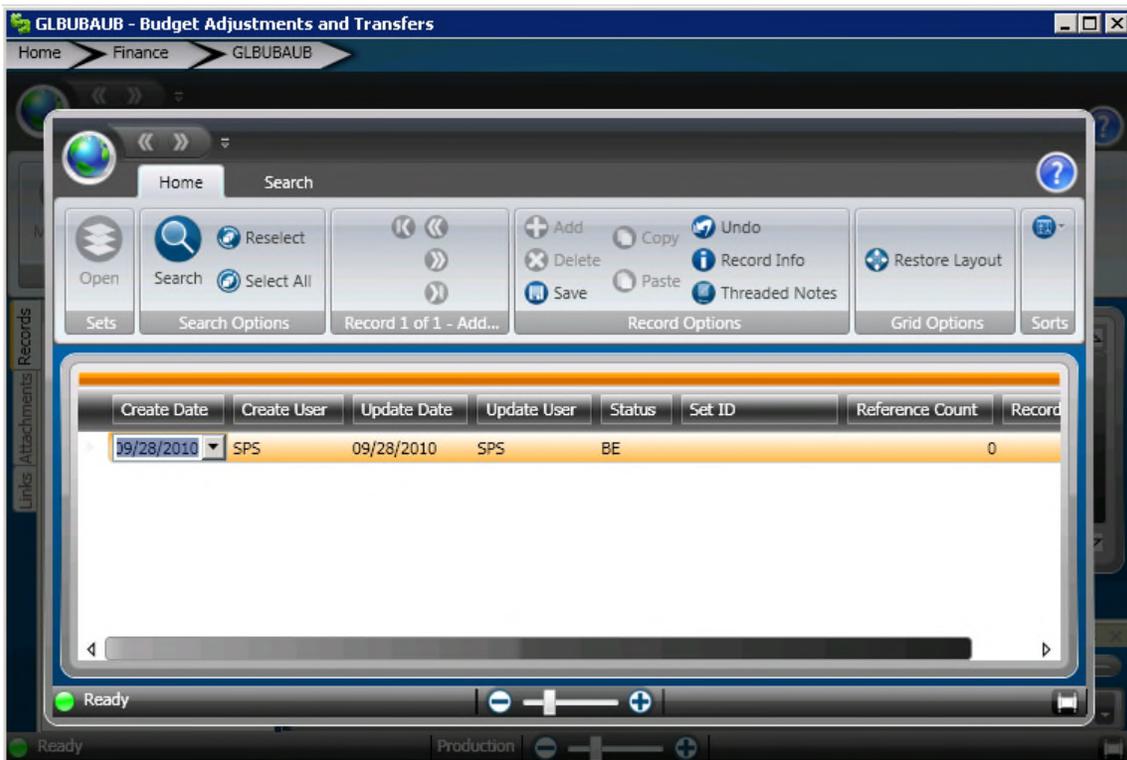


Figure 9: GLBUBAUB - Budget Batch Adjustment Screen - Set Tab

Batch Set ID's can be no more than eight characters and must begin with an alpha character. By now your organization has probably developed a standard naming convention for batch files. We recommend one character for the subsystem (B Budget Batch), two characters for the creator's initials, one character for the calendar year, one character for the month (1-9;Jan.-Sept.: A-C: Oct.-Dec.), two characters for the day (01-31), and finally a one character counter (A-Z). Alternatively a Seed number may be used (common code SYNO BTCHIDBU).

1.6.1.1.4 GLBUBAUB – Budget Adjustments tab

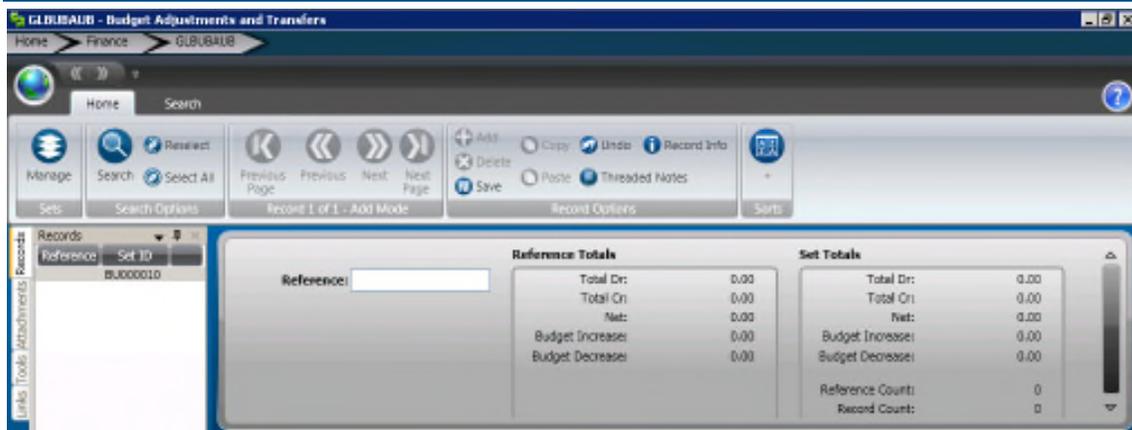


Figure 10: GLBUBAUB - Transaction Screen - Header

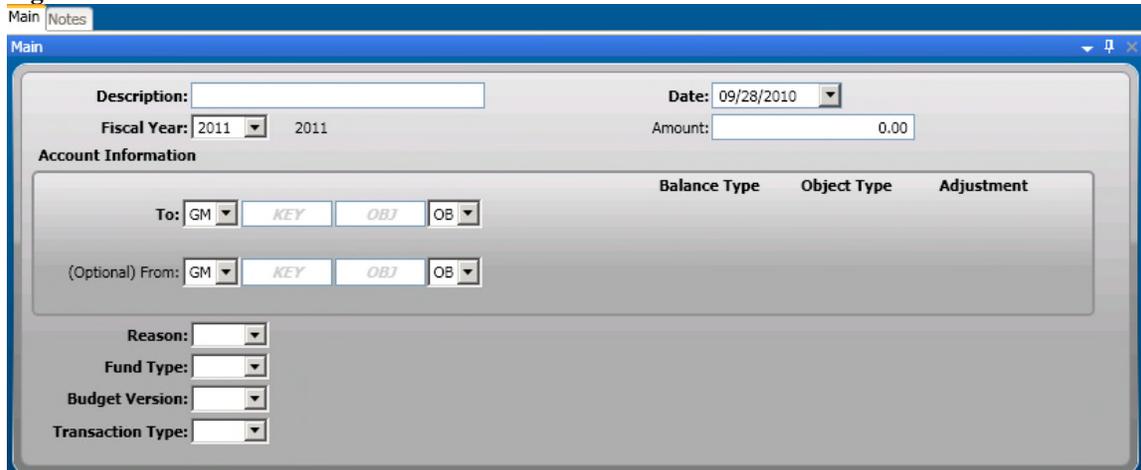


Figure 11: GLBUBAUB - Transaction Screen - Detail

Once we have created our batch the process for obtaining a batch proof is run with GLBUBABP and the posting distribution is launched with GLBUBADS

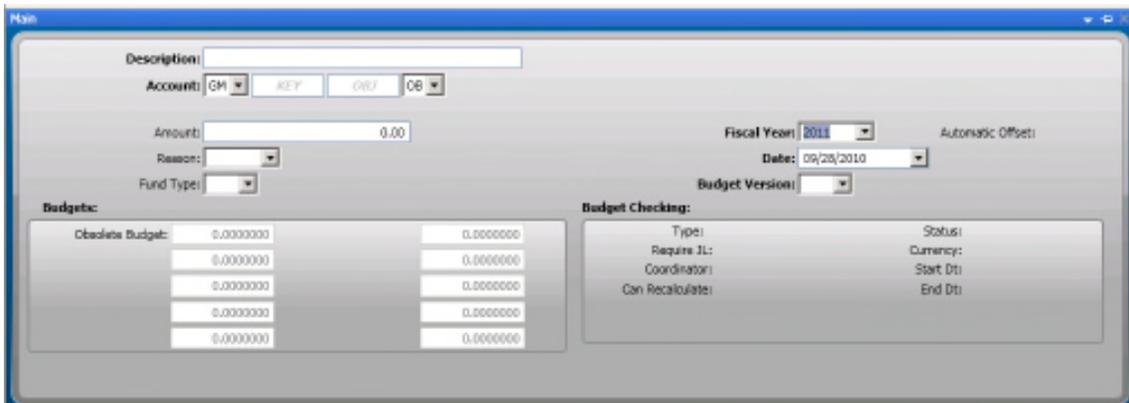
1.6.1.1.5 Intra-fund Balanced Budget Methods

The masks GLBUBAAN (Annual Budget Batch) and GLBUBAAJ (Adjustment Batch) allow Intra-fund Adjustments only and also are required to be in balance by debits and credits.

Something to keep in mind as we work through these screens is that they deal with budget amounts as debits and credits, to increase an expense account's budget the transaction is considered a credit, and to increase a revenue account's budget the transaction is considered a debit.

There are also fields for a Reason Code (Reason) and a Type Code (FT). These fields are validated against the same set of Common Codes as the fields from the Budget Batch Adjustment Screen.

1.6.1.1.6 GLBUBAAN – Budget Adjustments (Intra-Fund Annual Budget Batch)



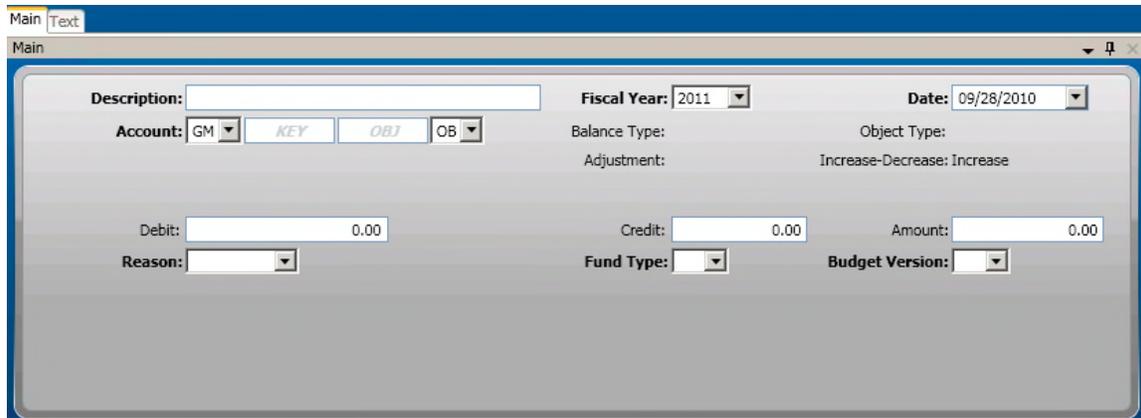
Budgets:

Obsolete Budget:	0.0000000	0.0000000
	0.0000000	0.0000000
	0.0000000	0.0000000
	0.0000000	0.0000000
	0.0000000	0.0000000

Budget Checking:

Type:	Status:
Require JL:	Currency:
Coordinator:	Start Dt:
Can Recalculate:	End Dt:

1.6.1.1.7 GLBUBAAJ – Budget Adjustments (Intrafund Budget Adjustment Batch)



The screenshot shows a software window titled "Main" with a "Text" tab. The interface contains several input fields and dropdown menus for configuring a budget adjustment batch. The fields are arranged in a grid-like layout:

- Description:** A text input field.
- Fiscal Year:** A dropdown menu set to "2011".
- Date:** A dropdown menu set to "09/28/2010".
- Account:** A series of dropdown menus for "GM", "KEY", "OBJ", and "OB".
- Balance Type:** A dropdown menu.
- Object Type:** A dropdown menu.
- Adjustment:** A dropdown menu.
- Increase-Decrease:** A dropdown menu set to "Increase".
- Debit:** A text input field with "0.00" entered.
- Credit:** A text input field with "0.00" entered.
- Amount:** A text input field with "0.00" entered.
- Reason:** A dropdown menu.
- Fund Type:** A dropdown menu.
- Budget Version:** A dropdown menu.

1.6.1.1.8 GLBUBAFI – Create Budget Batch from Flat File

If desired the file can be loaded into a batch process instead of directly updating the budget.

The file is then a bit more elaborate in that the user would be able to update the batch with descriptions, reason codes, funding type batch id's etc. The fields that are on the GLBUBAUB screen.

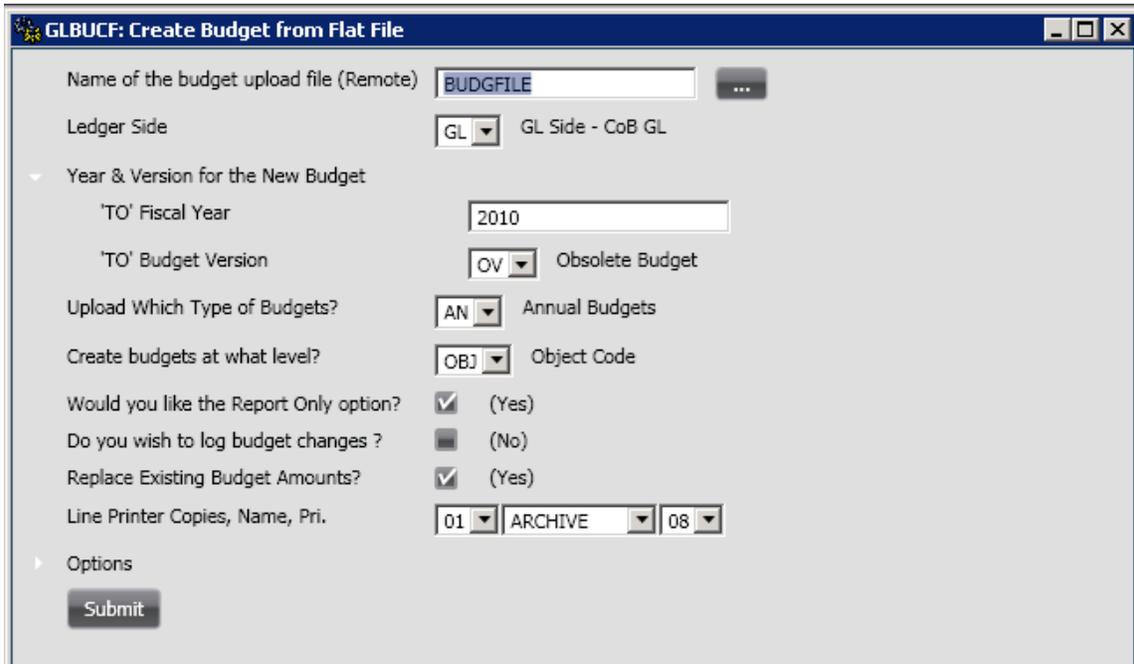
The instructions for this methodology can be found in the General Ledger User Guide in Section 4M.

1.6.1.1.9 GLBUCB – Copy Budgets

This function is used to copy and create budgets between named versions and the current or future fiscal years. When this function is requested, the system will display the selection criteria prompt.

Enter the four-digit base year, then enter the base budget named version and place the future year and named version in the following fields. The user has the option of copying at a percentage of the base year budget.

Also the budget can be copied as a report only option which essentially runs a trial pass.



GLBUCF: Create Budget from Flat File

Name of the budget upload file (Remote) ...

Ledger Side GL Side - CoB GL

Year & Version for the New Budget

'TO' Fiscal Year

'TO' Budget Version Obsolete Budget

Upload Which Type of Budgets? Annual Budgets

Create budgets at what level? Object Code

Would you like the Report Only option? (Yes)

Do you wish to log budget changes ? (No)

Replace Existing Budget Amounts? (Yes)

Line Printer Copies, Name, Pri.

Options

Figure 12: GLBUCB - Copy Budget Dialogue

1.7 BUDGET ITEM DETAIL

The Budget Item Detail process allows the user to create a full budget document in CDD using the variety of tables that are accessed from the Budget Item Detail Screens. There are a series of Budget Document screens and a series of Budget Item screens.

The Budget Document screens are used to create budget information for example departmental functions.

The Budget Item Screens are separated into four primary functions. The Position Budgeting screen BDPOSB, that interfaces from the Positioning Budget module. The BDMISC screen which is used for miscellaneous items, the BDEQPT screen which is used for Equipment items, and the BDTRVL screen which is used for the Travel Requests that departments may have for the year.

The Budget Item Detail Screens can be used selectively by the Budget Department to create a budget batch that updates the GLBUBAUB screen.

Here is an example of the BDMISC screen accessed from the mask BDUPDA.

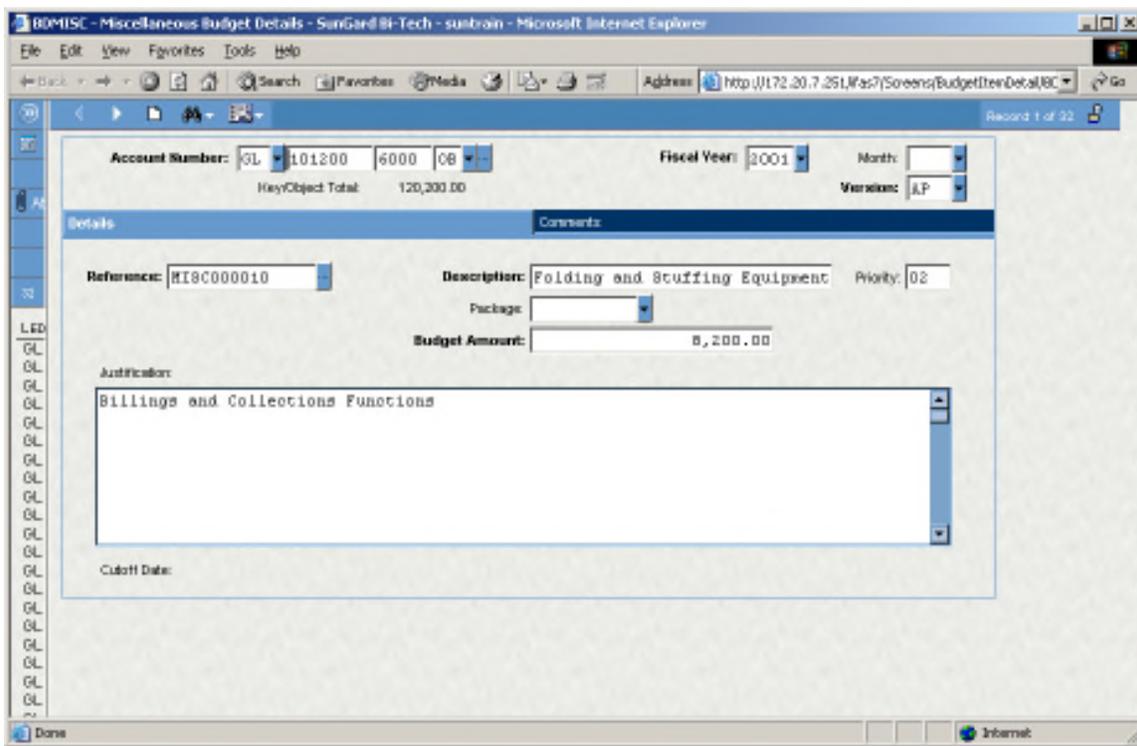


Figure 13: BDUPDAMISC - Budget Item Detail Entry screen

The implementation for Budget Item Detail is often put into Phase II of an implementation generally because the goal is to have the departments enter their own budget information.

Refer to separate budget item detail document.