CHICAGO BOOTH

CRSP®

Center for Research in Security Prices

DAILY US TREASURY DATABASE GUIDE

ASCII, EXCEL, SAS

Updated August 30, 2010

CHICAGO BOOTH

CRSP®

Center for Research in Security Prices

105 West Adams, Suite 1700
Chicago, IL 60603
Tel: 312.263.6400
Fax: 312.263.6430
Email: Support@crsp.ChicagoBooth.edu

CONTENTS

CHAPTER 1: INTRODUCTION

DESCRIPTION

The CRSP US Treasury Databases were developed by the Center for Research in Security Prices at the Booth School of Business, University of Chicago. CRSP provides complete historical descriptive information and market data including prices, returns, accrued interest, yields, and durations beginning in 1961.

The database is updated monthly and available in ASCII, Excel, and SAS formats on DVD.

DEVELOPMENT

Prices were manually input through December 31, 1989. Beginning January, 1990 through September, 1996, the prices were obtained from the Department of Commerce's electronic bulletin board (EBB). From October, 1996 through January 2009, prices were supplied by GovPX, Inc. ICAP began providing data in February 2009.

Manually recorded prices were double-entered, and programs were written to compare them. Once compared, price corrections were double-entered; the corrections were also compared for consistency. Several iterations of this process took place to arrive at the final, "clean" version of the file. Logical filters were then written and run to further clean the data.

Descriptive information and amounts outstanding were developed from the Monthly CRSP US Treasury Database.

SOURCES

DATA

Prior to January of 1962, price quotes were obtained from a number of different sources including First Boston, Morgan Guaranty, Bank & Quotation Record, New York Times, Government Actuary, and the Wall Street Journal. The 1920's and early 30's have a mixture of data from the New York Times and Government Actuary. By the late 1930's data are provided by Government Actuary and Bank & Quotation Record. In the early 1940's the predominant source is Government Actuary.   Salomon Brothers and a range of multiple sources including First Boston, Morgan Guaranty, Bank & Quotation Record, New York Times and the Wall Street Journal all provided data in the mid 1940's. By 1950, Salomon Brothers became the primary source and remained so through 1961.

Beginning with January of 1962, the majority of price quotes came from the Composite Closing Quotations for US Government Securities compiled by the Federal Reserve Bank of New York (FRBNY). In 1984, the quotation sheets were renamed the "Composite 3:30 P.M. Quotations for US Government Securities". The time at which the quotes were compiled was related to the fed wire deadline the FRBNY set for the transfer of securities. The deadline was set for 2:30 p.m. Eastern Time, but was regularly extended as much as three-quarters of an hour. The FRBNY trading desk began a "closing run" at 3:00 p.m. The reference to "closing quotations" from 1962 to 1984 refers to the "closing run" at the FRBNY. With the close of the day on October 15th, 1996 the FRBNY discontinued publication of composite quotations.

The start of the day, October 16, 1996, our source for price quotations, maturity dates, and coupon rates changed to GovPX, Inc. GovPX received its data from 5 inter-dealer bond brokers. Live, intra-day bids, offers and transactions in the active over-the-counter markets among these primary dealers were the source of GovPX's 5 p.m. End-of-day US Treasury price quotes.

GovPX provided data until it was acquired by ICAP. Beginning in February 2009, CRSP began releasing the daily and monthly treasury databases using the ICAP data.

The amount outstanding debt is obtained from the *Monthly Statement of the Public Debt of the United States* published by the Treasury Department. The amount of publicly held debt is obtained from the quarterly *US Treasury Bulletin*. Money Rates are obtained from the Federal Reserve. Issue date, coupon payable dates, bank eligibility, tax status and call status are obtained from the US Treasury Department.

Prior to 1990, CUSIP was obtained from *Standard & Poor's CUSIP Directory*. From January, 1990 through October 15th, 1996, the CUSIP was obtained from the Composite 3:30 p.m. quotations for US Government Securities. GovPX, beginning October 16, 1996, provided the CUSIP number. Beginning in February 2009, the CUSIP number is provided by ICAP. When in question, the CUSIP is verified by *Standard & Poor's CUSIP Directory*.

SPREADS DIFFERENCES OVER TIME

The FRBNY described its listed bid price as "...the most widely quoted price from the range of quotations received". The ask price was determined by the FRBNY based on what they expect a typical bid-ask spread to be. The rule used to make this derivation was not public domain.

GovPX described its listed bid and ask prices as the "best price". To determine their "best price" they observed the prices from the 5 inter-dealer brokers and reported the bid and ask prices that produced the smallest bid-ask spread. This practice resulted in stable spreads that showed little to no fluctuation over time.

ICAP provides the actual bid and ask quotes and calculates real spreads. The result is spreads that fluctuate daily. A seam in the spread data is observed in February 2009 with the change in data sources and the noticeable increase in fluctuation. Regardless of sources, the midpoints of the imputed and real spreads are very close.

All data are checked for internal consistency with each release of the file. Secondary sources, such as the *Wall Street Journal*, are used to check suspect prices.

DIFFERENCES BETWEEN DAILY AND MONTHLY FILES

The CRSP Daily US Treasury Files are a superset of the CRSP Monthly US Treasury Files with four exceptions.

1. When-issued prices are included in the Daily Files. All prices before an issue's dated date can be identified as when-issued prices.
2. Government Certificate of Deposit, Commercial Paper, and Federal Funds rates are included in the daily files.
3. Bond indexes equivalent to the four Fama Files in the monthly database have not yet been developed for the daily database.
4. C and FORTRAN programming access is provided for the daily data files, and FORTRAN only for the monthly data files.

Certain derived data items are not stored, but can be accessed using the utility functions that are provided. Other less frequent data are only stored on the observation dates. See Section 4 for information on accessing the daily data.

NOTATIONAL CONVENTIONS

- All data items and names that occur within FORTRAN or C programs are printed using a `constant - width (courier)` font. These names can be variable names, parameter names, subroutine names or keywords. For example, CUSIP refers to the CUSIP Agency identifier, while CUSIP refers to the variable that the programs use to store this identifier.
- Names of FORTRAN common blocks are delimited by slashes (/ /).

CHAPTER 2: DATABASE STRUCTURE

The Daily CRSP US Treasury Database consists of three primary files: the Calendar File, the Master File, and the Cross-Sectional File. These are supplemented by the derived CRSP Fixed Term Indices Files.

The files are organized both as time series by issue and cross-sectionally by date.

Diagrams are provided as follows:

- The Calendar File,
- The Master File,
- The Cross-Sectional File, and
- The Fixed Term Indices Files.

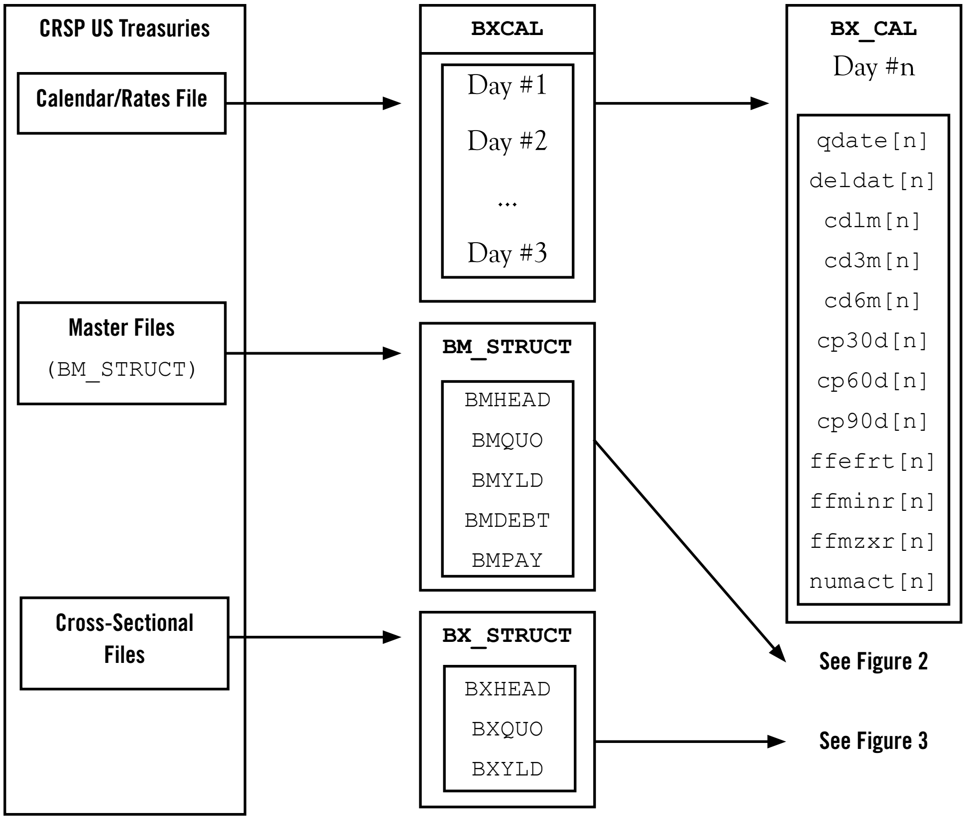
See Chapter 3 for the available data items and their descriptions.

See Chapter 4 for file specifications.

CALENDAR FILE

The Calendar File contains Daily Quote Dates and Delivery Dates as well as several Julian, linear, and other date information derived from these values.

FIGURE 1: CALENDAR FILE STRUCTURE

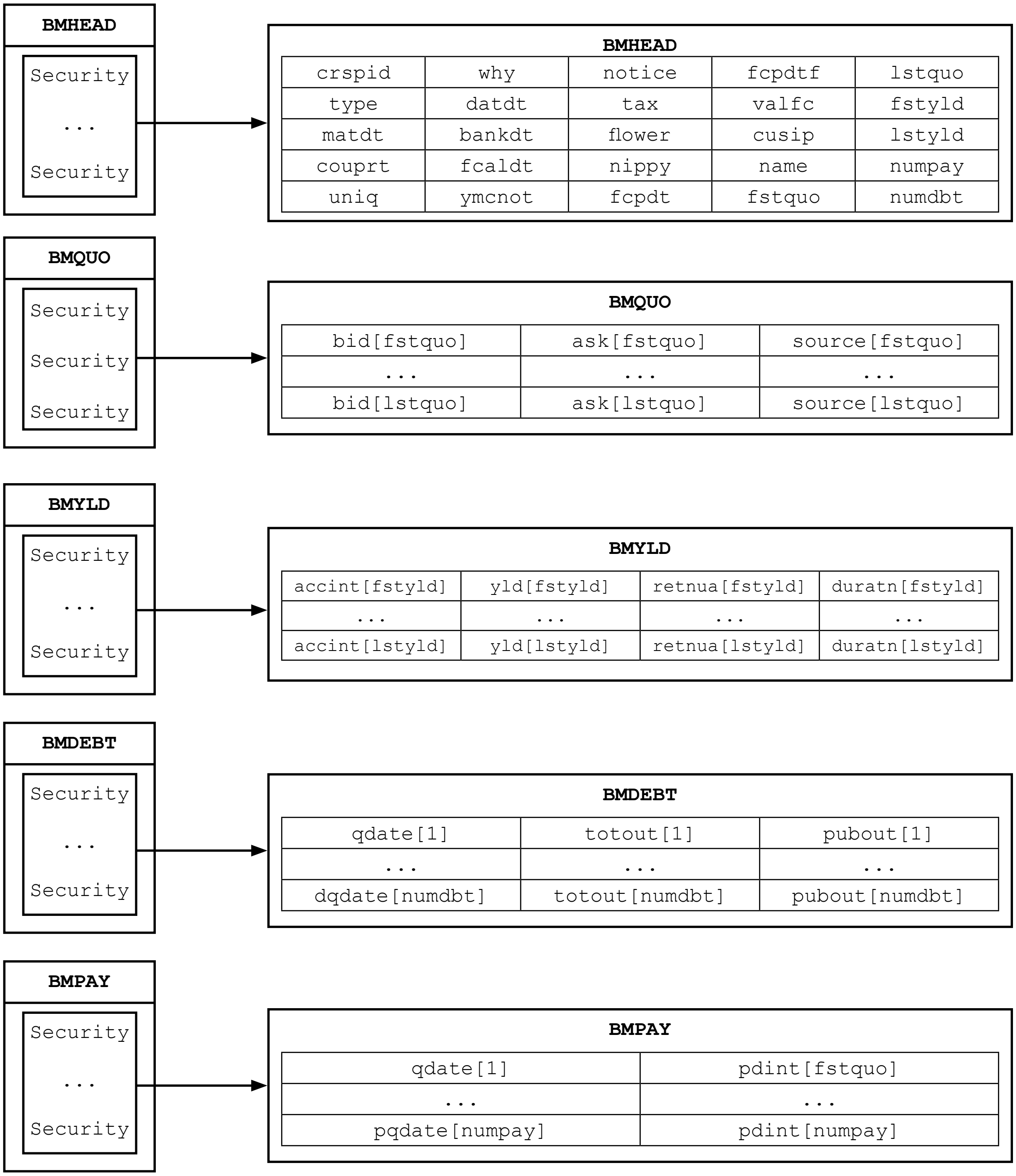


MASTER FILE

The Master File (MBM) contains end-of-day price data on virtually all negotiable direct obligations of the United States Treasury for the period June 14, 1961, to the present. The Master File is sorted by issue.

File sets are separated into three categories: header information, raw daily data, and derived daily data. Header information contains CRSP identifiers and characteristics set by the US Treasury; interest payment dates, callable status, data ranges on quotes, number of amounts outstanding, and number of interest payments.

FIGURE 2: MASTER FILE STRUCTURE



BMHEAD

crspid	why	notice	fcpdtf	lstquo
type	datdt	tax	valfc	fstyld
matdt	bankdt	flower	cusip	lstyld
couprt	fcaldt	nippy	name	numpay
uniq	ymcnot	fcpdt	fstquo	numdbt

BMQUO

bid[fstquo]	ask[fstquo]	source[fstquo]
...	...	...
bid[lstquo]	ask[lstquo]	source[lstquo]

BMYLD

accint[fstyld]	yld[fstyld]	retnua[fstyld]	duratn[fstyld]
...	...	...	...
accint[lstyld]	yld[lstyld]	retnua[lstyld]	duratn[lstyld]

BMDEBT

qdate[1]	totout[1]	pubout[1]
...	...	...
dqdate[numdbt]	totout[numdbt]	pubout[numdbt]

BMPAY

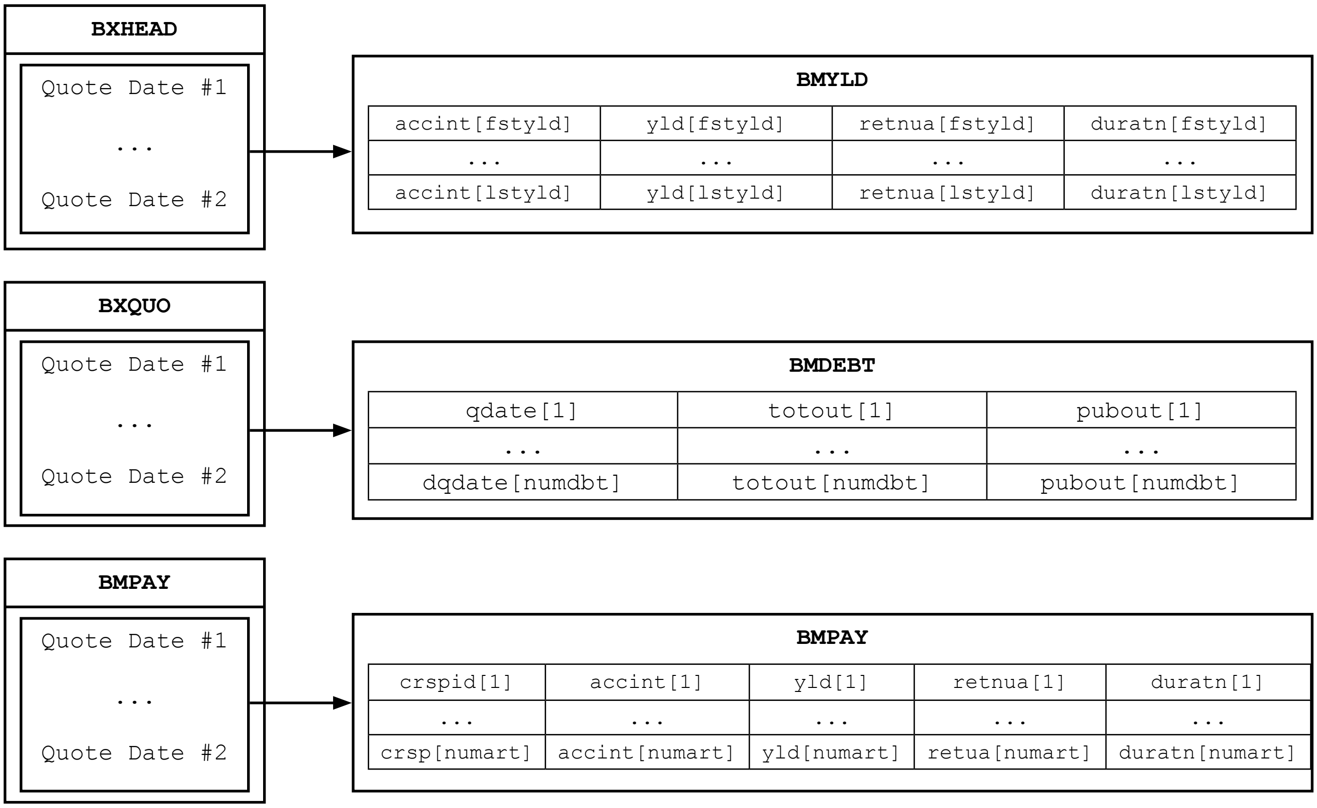
qdate[1]	pdint[fstquo]
...	...
pqdate[numpay]	pdint[numpay]

CROSS-SECTIONAL FILES

The Cross-Sectional File (MXM) contains the same information as the Master File, except it is sorted by Quote Date instead of by issue. Section 3 contains detailed descriptions of the data variables.

File sets are separated into three categories: header information, raw daily data, and derived daily data. Header information contains CRSP identifiers and characteristics set by the US Treasury; interest payment dates, callable status, data ranges on quotes, number of amounts outstanding, and number of interest payments.

FIGURE 3: CROSS-SECTIONAL FILE STRUCTURE


BXHEAD
Quote Date #1
...
Quote Date #2
BMYLD
accint[fstyld] yld[fstyld] retnua[fstyld] duratn[fstyld]
... ...
accint[lstyld] yld[lstyld] retnua[lstyld] duratn[lstyld]
BXQUO
Quote Date #1
...
Quote Date #2
BMDEBT
qdate[1] totout[1] pubout[1]
... ...
dqdate[numdbt] totout[numdbt] pubout[numdbt]
BMPAY
Quote Date #1
...
Quote Date #2
BMPAY
crspid[1] accint[1] yld[1] retnua[1] duratn[1]
... ...
crsp[numart] accint[numart] yld[numart] retua[numart] duratn[numart]

CRSP FIXED TERM INDICES FILE

This set of derived files offers 30-, 20-, 10-, 7-, 5-, 2- and 1-year target maturity indices, sorted by term type and quote date. This index creates a sophisticated bond yield curve, allowing the data items to be referenced by returns, prices, and duration. Start dates vary based upon term types selected. Programming support is not provided for the CRSP Fixed Term Indices File.

The Fixed Term Indices File contains a variable number of data records for each quotation date and term type. There are no sample programs available for this file.

TERMTYPE	QDATE	CRSPID	YEARSTM	RETADJ	YTM	ACCINT	DURATN	BID	ASK
–	–	–	–	–	–	–	–	–	–
TERMTYPE[1]	QDATE[N]	CRSPID [N]	YEARSTM[N]	RETADJ[N]	YTM[N]	ACCINT[1]	DURATN[N]	BID[N]	ASK[N]
TERMTYPE[2]	QDATE[1]	CRSPID [1]	YEARSTM[1]	RETADJ[1]	YTM[1]	ACCINT[1]	DURATN[1]	BID[1]	ASK[1]
–	–	–	–	–	–	–	–	–	–
TERMTYPE[N]	QDATE[N]	CRSPID [N]	YEARSTM[N]	RETADJ[N]	YTM[N]	ACCINT[N]	DURATN[N]	BID[N]	ASK[N]

CHAPTER 3: DATA DEFINITIONS

This section describes the data items provided within our files. Each description is preceded with a line containing three bolded items:

- The Variable Name
- A Short Description of the Data Represented
- Data Types

The data items in this section are grouped logically according to six data types:

1. CALENDAR - Trading Calendar and Government Rates

2. HEADER - Issue Identification, Characteristics, and Data Ranges

3. QUOTES - Raw Pricing Data

4. YIELDS - Derived Yields, Duration, Returns, and Accrued Interest

5. DEBT - Amounts Outstanding

6. PAYMENTS - Interest Payments

Certain data types are organized by both issue and date (See the figures in Chapter 2). More complete information on accessing the data items using variables in CRSP FORTRAN and C programs is contained in Chapter 4.

Information on the Fixed Term Indices File is available in this chapter.

1. CALENDAR - CALENDAR AND GOVERNMENT RATES

The BXCAL structure contains the trading calendar and summary information for each date in the CRSP US Treasury Database. The three types of information include:

1. Trading calendar quote dates and delivery dates.
2. Government rates for certificates of deposit, commercial paper, and federal funds.
3. Number of trading US Government securities.

QDATE **Date of Quotation, in YYYYMMDD Format** **integer**

QDATE contains the trading quote dates for the files. These dates are stored in form YYYYMMDD (year, month, and date).

DELDAT Delivery Date, in YYYYMMDD Format `integer`

DELDAT contains the delivery date for a corresponding quote date. These dates are stored in the form YYYYMMDD (year, month, date).

The Federal Reserve Bank of New York, the source from January 1962 through October 15, 1996, assumed cash transactions on delivery date. The delivery date usually fell two business days after the quotation date. GovPX, the source from October 16, 1996, reports delivery data the next business day after the end quote date.

CD1M One-Month Certificate of Deposit Rate `real`

Certificate of deposit rate is the average of secondary market morning offering rates for time certificates of deposit of major money market banks. A Certificate of Deposit is an unsecured note issued by companies for short-term borrowing purposes.

CD3M Three-Month Certificate of Deposit Rate `real`

CD6M Six-Month Certificate of Deposit Rate `real`

CP30D 30-Day Commercial Paper Rate `real`

Commercial paper rate is an average of posted 10 a.m. offering rates of five dealers. Rates are quoted on a discount basis. It is an unsecured note issued by companies for short-term borrowing purposes. Commercial paper is frequently sold by the issuer directly to the investor, the latter normally being institutions, money-market funds, insurance companies, corporations, bank trust departments and pension funds. Commercial paper is also placed by intermediary banks or securities dealers.

CP60D 60-Day Commercial Paper Rate `real`

CP90D 90-Day Commercial Paper Rate `real`

FFEFRT Federal Funds Effective Rate `real`

The effective rate is a weighted average of the rates on overnight federal funds transactions arranged by federal funds brokers. The Federal Funds Rate is the rate of interest charged on federal funds loaned by and to commercial banks and is regarded by the Federal Reserve System's regulatory authorities as an important determinant of bank liquidity.

FFMINR Federal Funds Minimum Trading Range `real`

FFMAXR Federal Funds Maximum Trading Range `real`

NUMACT Number of Active Issues `integer`

The number of active US Treasury issues quoted on a quotation date.

2. HEADER - ISSUE IDENTIFICATION, CHARACTERISTICS, AND DATA RANGES

This structure contains header information for issues. There are three types of information included:

1. Identification assigned by CRSP or CUSIP to uniquely identify the issue.
2. Characteristics of the issue set by the treasury, such as interest dates and callable status.
3. Data ranges, including the date ranges of quotes, the number of amounts outstanding, and the number of interest payments.

CRSPID **CRSPID (CRSP Assigned Unique Issue Identification Number)** **character*15**

The CRSPID is in the format YYYYMMDD.TCCCCE, where:

YYYY	=	Maturity Year	
MM	=	Maturity Month	
DD	=	Maturity Day	
T	=	Type Of Issue (TYPE)	
CCCC	=	Integer Part of	(COUPRT x 100)
E	=	Uniqueness Number	(UNIQ)

For example, 19850515.504250 identifies a 4 1/4% callable bond which matures May 15, 1985. For callable notes and bonds, the YYYY portion of the CRSPID contains only the final maturity date of the issue and not the first eligible call date for that issue.

The variable CRSPID is a composite of other variables. Mathematical operations to retrieve parts of the CRSPID are unnecessary when using the Master File.

TYPE **Type of Issue** **integer**

1	=	Noncallable bond	
2	=	Noncallable note	
3	=	Certificate of indebtedness	
4	=	Treasury Bill	
5	=	Callable bond	
6	=	Callable note	
7	=	Tax Anticipation Certificate of Indebtedness	
8	=	Tax Anticipation Bill	
9	=	Other – this flags issues with unusual provisions.	See Appendix A

MATDT **Maturity Date at Time of Issue, in YYYYMMDD Format** **integer**

COUPRT Coupon Rate (percent per annum) **real*8**

UNIQ Uniqueness Number **integer**

Uniqueness number assigned to CRSPID if maturity date, coupon rate and type are not sufficient enough to distinguish between two securities; 0 otherwise.

WHY Reason for End of Data on File **integer**

0 = Still quoted on last update of file

1 = Matured

2 = Called for redemption

3 = All exchanged

4 = Sources no longer quote issue

DATDT Date Dated by Treasury, in YYYYMMDD Format **integer**

Coupon issues accrue interest beginning on the dated date. This may result in a modified first coupon payment if the dated date is not a regular interest payment date.

DATDT is 0 if it is not available or not applicable, as is the case with Treasury bills.

BANKDT Bank Eligibility Date at Time of Issue, in YYYYMMDD Format **integer**

The earliest date at which a security is to become "bank eligible". A security is bank eligible if a bank can own it. Some 2 ½%'s and 2 ¼%'s issued during and immediately after WWII limited negotiability because of prohibitions and restrictions on bank ownership.

0 = no restrictions apply

YYYYMMDD = restrictions removed or scheduled to have been removed on this date

All remaining restrictions were removed on January 1, 1955. The last bank eligible CRSPID in the file is dated November 15, 1945 and matured on December 15, 1972.

FCALDT First Eligible Call Date at Time of Issue, in YYYYMMDD Format **integer**

FCALDT is 0 if the security is not callable.

YMCNOT Year and Month of First Call Notice, in YYYYMMDD Format **integer**

YMCNOT is 0 if not called or not callable.

NOTICE Notice Required on Callable Issues **integer**

TAX **Taxability of Interest** **integer**

1 = Fully taxable for federal income tax purposes

2 = Partially tax exempt, i.e. interest on first $3000 of tax exempt bonds of this class, at par value, are subject to surtax but not to normal tax

3 = Wholly tax exempt

FLOWER **Payment of Estate Tax Code** **integer**

1 = No special status

2 = Acceptable at par and accrued interest if owned by decedent at time of death; a flower bond

3 = Acceptable at par and accrued interest if owned by decedent during entire 6 month period preceding death; a flower bond

NIPPY **Number of Interest Payments Per Year** **integer**

0 = Treasury bill or certificate paying interest only at maturity

1 = Annual interest

2 = Semi-annual interest

4 = Quarterly interest

All interest-bearing negotiable Treasury securities issued since the beginning of WWI have paid interest semi-annually. The last outstanding issue that paid interest quarterly was the Panama Canal Loan 3%'s due June 1, 1961.

FCPDT **First Coupon Payment Date, in YYYYMMDD Format** **integer**

FCPDT is 0 if not applicable. FCPDTF indicates whether the first coupon date is an estimate or a verified date.

FCPDTF **First Coupon Payment Date Flag**

integer

0 = Treasury bill or not applicable

-1 = First coupon date is estimated from the normal coupon payment cycle

1 = First coupon date has been verified on the Treasury Offering Circular

VALFC **Amount of First Coupon Per $100 Face Value** **real*8**

CUSIP **CUSIP Number** **character*8**

A CUSIP number (Committee on Uniform Securities Identification Procedures) is an identifying number assigned to a publicly-traded security. A nine-digit code is permanently assigned to each issue and is generally printed on the face of the security if it is in physical form. The first eight digits are included in the CRSP file. The ninth digit is a check digit derived from the first eight digits. Missing CUSIPs are assigned the value OXX. The earliest maturity date on file with a CUSIP is February 15, 1969.

NAME **Name of Government Security** `character*8`

NAME	ITYPE	EXPLANATION
BILL	4	
T_A_BILL	8	Tax Anticipation
T_A_CTF	7	Tax Anticipation
BOND	1,5,9	
CNV_BOND	1	Convertible
CONSOL	9	Consol
CTF	3,7,9	Certificate of Deposit
NOTE	0,2,6,9	
1LL_BOND	5	First Liberty Loan
1LL_CV	5	1LL First Conversion
1LL_2CNV	5	1LL Second Conversion
2LL_BOND	5	Second Liberty Loan
2LL_CNV	5	2 LL First Loan Conversion
3LL_BOND	1	Third Liberty
4LL_BOND	9	Fourth Liberty Loan
4LL_CALL	9	Fourth Liberty Loan called
PCL_BOND	1,5	Panama Canal Loan

FSTQUO **Day Number of Issue's First Quote on File** `integer`

The QDATE array can be used to translate day numbers into YYYYMMDD format dates.

LSTQUO **Day Number of Issue's Last Quote** `integer`

The QDATE array can be used to translate day numbers into YYYYMMDD format dates. An issue that matures typically stops trading on the first quote date with a delivery date greater than or equal to the issue's maturity date.

FSTYLD **Day Number of Issue's First Yield** `integer`

The QDATE array can be used to translate day numbers into YYYYMMDD format dates.

LSTYLD **Day Number of Issue's Last Yield** `integer`

The QDATE array can be used to translate day numbers into YYYYMMDD format dates. An issue that matures typically stops trading on the first quote date with a delivery date greater than or equal to the issue's maturity date.

NUMPAY **Number of Interest Payments** `integer`

Count of observations in BMPAY structure.

NUMDBT **Number of Amount Outstanding Observations** `integer`

Count of valid observations in the BMDEBT structure.

3. QUOTES - RAW DATA

CRSP-generated data, such as yield and duration, are calculated from secondary market cash transaction prices. CRSP derives its data from the bid and ask prices. CRSP data are calculated based on cash transactions on the quotation date. CRSP's primary data sources assume cash transactions on delivery date. Quotes from the Federal Reserve Bank of New York usually have a delivery date two business days after the quotation date. Quotes from GovPX, Inc. usually have a delivery date one business day after the quotation date. The delivery date usually falls two business days after the quotation date. CRSP takes this into account when verifying the internal consistency of the files.

When-issued prices are included in the file when quoted. Any price with a quote date before an issue's dated date is classified as when-issued.

Quotes are present in the Master and Cross-Sectional files. In the Master File, the quotes are sorted by issue, then date. For any issue, header variables FSTQUO and LSTQUO are used to delimit the number of days within the range. In the Cross-Sectional File, the quotes are sorted by date, then issue. For any quote date, calendar variable NUMACT contains the number of quotes available.

CRSPID **CRSPID (CRSP Assigned Unique Issue Identification Number)** **character*15**

See CRSPID on page 8

QDATE **Date of Quotation in YYYYMMDD Format** **integer**

See QDATE on page 6.

BID & ASK **Prices** **real*8**

The bid price is the price at which a buyer is willing to purchase a security. The ask price is the price at which the seller is offering to sell the security.

Arrays BID and ASK contain end-of-day bid and ask information, when available, for each quote date prior to maturity. If BID and ASK are not available, whatever quote information is available is used and coded using the following conventions:

INFORMATION IN DATA SOURCE	BID	ASK
Bid and Ask	Bid	Ask
Mean of Bid and Ask	Mean	Mean
Bid only	Bid	-Bid
Ask only	-Ask	Ask
Sale (last trading price)	Sale	0
No price Sale	0	0

SOURCR **Primary Data Source** **character*1**

R = Federal Reserve Bank of New York

S = Salomon Brothers

W = Wall Street Journal - present (Associated Press: 6/14/61-8/20/87, Bloomberg: 8/28/87-7/2/90, Bear-Stearns: 12/4/90-2008)

M = No quote was available

X = GovPX, Inc.10/1996-1/2009, ICAP 2/2009-present

4. YIELDS - DERIVED DATA

For bonds that have been called, or are likely to be called, the original maturity date is no longer valid for computing duration and yield. In these cases the anticipated call date is used as the working maturity date.

The following note applies to the promised daily yield (YIELD) and duration (DURATN) variables.

STATUS	YIELD AND DURATION COMPUTED TO
Called	Next call date
Callable and priced at a premium	Next call date
Callable and priced at a discount	Maturity date
Not callable	Maturity date

Users should be cautious in interpreting yields based on issues close to maturity. Quotes on these instruments are not always reliable due to infrequent trading.

Yields are present in the Master and Cross-Sectional files. In the Master File, the yields are sorted by issue, then date. For any issue, header variables FSTQUO and LSTQUO can be used to delimit the number of days within the range. In the Cross-Sectional File, the yields are sorted by date, then issue. For any quote date, calendar variable NUMACT contains the number of yields available.

CRSPID **CRSPID (CRSP Assigned Unique Issue Identification Number)** **character*15**

See CRSPID on page 8

QDATE **Date of Quotation in YYYYMMDD Format** **integer**

See QDATE on page 6.

ACCINT **Total Accrued Interest at End of Day** **real*8**

Accrued interest on U.S. Treasury marketable securities is calculated on the basis of the number of days between interest payment dates for a $100 bond or note. Interest is accrued either from the last interest payment date or from the dated date (when an interest payment has not yet occurred) to the quotation date.

YLD **Promised Daily Yield** **real*8**

YLD is the promised yield daily rate, also called daily yield to maturity.

On any given date, the promised yield of a security is the single interest or discount rate that makes the sum of the present values of the principal at maturity plus future interest payments equal to the full price of the security. The full price is the nominal price plus the accrued interest. If a price is missing, the YLD is set to -99.

RETNUA **Unadjusted Return** **real*8**

RETNUA is price change plus interest, divided by last day's price. It is set to a large negative number for days in which a return cannot be calculated, i.e. if the price is missing for either this day or last day. Missing returns are set to -99.

$RETNUA = \frac{XNUM}{XDEN}$, where

When BID and ASK available:

$$XDEN = \frac{BID(I-1)+ASK(I-1)}{2} + ACCINT(I-1)$$

$$XNUM = \frac{BID(I)+ASK(I)}{2} - \frac{BID(I-1)+ASK(I-1)}{2} + YINT$$

$$YINT = PDINT(I) + ACCINT(I) - ACCINT(I-1)$$

For all other cases:

```
XNUM = BID(I) - BID(I-1) + YINT

XDEN = BID(I-1) + ACCINT(I-1)

YINT = PDINT(I) + ACCINT(I) - ACCINT(I-1)
```

DURATN **Duration (Macaulay's Duration)** **real*8**

Duration is the weighted average number of days until the cash flows occur, where the present values, discounted by yield to maturity, of each payment are used as the weights. Also known as Macaulay's Duration.

If, P_{t_0} , , ..., P_{t_n} are the present values at time t_0 of payment promised at perhaps unequally spaced time intervals *t1*, *t2*, ..., *tn* then the duration of that promised stream measured at t_0 is:[1]

$$D_{t_0} = \frac{\sum_{j=1}^{j=n}(t_j - t_0)P_{t_j}}{\sum_{j=1}^{j=n}P_{t_j}} = \frac{\sum_{j=1}^{j=n}t_j P_{t_j}}{\sum_{j=1}^{j=n}P_{t_j}} - t_0$$

5. DEBT - AMOUNTS OUTSTANDING

Amounts outstanding are present in the Master File, sorted by issue and date. The header variable NUMDBT contains the number of records available for an issue. These values are typically reported monthly. Total amounts outstanding are obtained from the Monthly Statement of the Public Debt of the United States. The amounts publicly held are obtained from the quarterly Treasury Bulletin. Before 1983, the *Treasury Bulletin* was reported monthly.

`CRSPID` **CRSPID (CRSP Assigned Unique Issue Identification Number)** `character*15`

See CRSPID on page 8

`DQDATE` **Effective Date of Amount Outstanding Values in YYYYMMDD Format** `integer`

`TOTOUT` **Face Value Outstanding** `integer`

Amount (face value) issued and still outstanding in millions of dollars. Set to 0 for unknown values up to December 31, 1961 and set to -1 for unknown values thereafter.

`PUBOUT` **Publicly Held Face Value Outstanding** `integer`

Amount (face value) held by the public in millions of dollars. This is the total amount outstanding (TOTOUT) minus the amount held in U.S. Government accounts and Federal Reserve Banks. This amount is not available for Treasury Bills and is always set to 0. For other issues, set to 0 for unknown values up to December 31, 1961 and set to -1 for unavailable values after December 31, 1961. After December 31, 1982, these numbers are reported quarterly instead of monthly, and the reported values are carried forward the next two months.

6. PAYMENTS - INTEREST PAYMENTS

Payments are present in the Master File, sorted by issue and date. The values are derived from the frequency and amount of coupon payments, the first coupon date, value of first coupon, and maturity date. Payments are only stored for the time range of an issue's quotes. Bills have no payment records.

`CRSPID` **CRSPID (CRSP Assigned Unique Issue Identification Number)** `character*15`

SSee CRSPID on page 8

`PQDATE` **Interest Payment Dates, in YYYYMMDD Format** `integer`

`PDINT` **Interest Paid** `real*8`

PDINT is the coupon payable on the interest payment date.

CRSP FIXED TERM INDICES FILES

The Fixed Term Indices Files contain 1, 2, 5, 7, 10, 20 and 30 year Fixed Term Indices. These issues are sorted by termtype, which distinguishes the length of maturity. A valid issue that best represents each term is chosen at the end of each month for each of the above referenced fixed terms. A valid issue is one that is at least one half year prior to the target maturity date and is fully taxable. The selection process filters a representative bond from each of the fixed term groups. The first selection criteria are; a non-callable, non-flower bond that is closest to the target maturity of its group and fully taxable. If more than one issue remains, and/or none are available which fit the above criteria, they are then respectively filtered on the basis of flower bonds acceptable at par, and accrued interest if owned by descendent at time of death.

These values were designed to plot a sophisticated yield curve, and the user may reference the yields with returns, prices, and durations.

Data for the Fixed Term Indices Daily Files begins June 14, 1961. Maturities are as follows:

TERMTYPE	INDEX
3012	30 year
2012	20 year
1012	10 year
712	7 year
512	5 year
212	2 year
112	1 year

INDICES VARIABLE ITEMS

ACCINT **Total Accrued Interest at End of Day** **real*8**

Accrued interest on U.S. Treasury marketable securities is calculated on the basis of the number of days between interest payment dates for a $100 bond or note. Interest is accrued either from the last interest payment date or from the dated date (when an interest payment has not yet occurred) to the quotation date.

BID & ASK **Prices** **real*8**

The bid price is the price at which a buyer is willing to purchase a security. The ask price is the price at which the seller is offering to sell the security.

Arrays **BID** and **ASK** contain day-end bid and ask information, when available, for each quote date prior to maturity.

INFORMATION IN DATA SOURCE	BID	ASK
No price	0	0
Sale	Sale	0
Bid only	Bid	-Bid
Ask only	-Ask	Ask
Bid and Ask	Bid	Ask
Mean of Bid and Ask	Mean	Mean

CRSPID **CRSPID (CRSP Assigned Unique Issue Identification Number)** **character*15**

The CRSPID is in the format **YYYYMMDD.TCCCCE**, where:

YYYY = Maturity Year

MM= Maturity Month

DD = Maturity Day

T = Type Of Issue (TYPE)

CCCC = Integer Part of (COUPRT x 100)

E = Uniqueness Number (UNIQ)

For example, 19850515.504250 identifies a 41/4%callable bond which matures May 15, 1985. For callable notes and bonds, the YYYY portion of the CRSPID contains only the final maturity date of the issue and not the first eligible call date for that issue.

DURATN Duration (Macaulay's Duration) real*8

Duration is the weighted average number of days until the cash flows occur, where the present values, discounted by yield to maturity, of each payment are used as the weights. Also known as Macaulay's Duration.

If P_{t_0}, P_{t2},... , P_{t_n} are the present values at time t_0 of payment promised at perhaps unequally spaced time intervals *t1, t2, ..., tn* then the duration of that promised stream measured at t_0 is:

$$D_{t_0} = \frac{\sum_{j=1}^{j=n}(t_j - t_0)P_{t_j}}{\sum_{j=1}^{j=n}P_{t_j}} = \frac{\sum_{j=1}^{j=n} t_j P_{t_j}}{\sum_{j=1}^{j=n}P_{t_j}} - t_0$$

QDATE Date of Quotation, in YYYYMMDD Format integer

QDATE contains the Trading Quote Dates for the Bond Files. These dates are stored in the form YYYYMMDD (year, month, and date).

RETADJ Daily Holding Period Return real*8

RETADJ is the daily holding period return expressed as a percentage.

RETADJ(I)=100**RETNUA(I)*

TERMTYPE Index Identification Number integer

Fixed term index identification number links all results in the Fixed-Term Indices File. The identification is typically in the form YYYYMM, where YYYY is the number of years to maturity of issues selected in the index and MM is the number of months an issue is held once selected before another is chosen.

YEARSTM Years to Maturity real*8

Number of years left to maturity. In the fixed term index files, YEARSTM contains the time left to maturity of the selected issue as of the quote date, expressed annually as a decimal amount.

YTM Annualized Yield real*8

YTM is the annualized YIELD to maturity expressed as a percent per annum. See YIELDS: YIELD.

YTM(I)=100**[YLD(I)**365*]*

[1] *Some Theoretical Problems of Interest Rates, Bond Yields and Stock Prices in the United States Since 1856.* Frederick R. MacAulay, National Bureau of Economic Research, 1938, 44-53.

[1] Coping with the Risk of Interest-Rate Fluctuations: Returns to Bondholders from Naive and Optimal Strategies, Lawrence Fisher and Roman L. Weil, Journal of Business, vol. 44, 415.

CHAPTER 4: ACCESSING THE DATA

The Daily CRSP US Treasury Database is available in ASCII, Excel, and SAS.

1. The ASCII (text) files are the basis from which all other data formats are built. C and Fortran sample programs are provided to access the ASCII formats. See Section 4.3 for details about the ASCII file specifications for the Master (BM) File, the associated Header File, Cross Sectional (BX) File and the Fixed Term Indices File. Section 4.2 contains descriptions of the sample programs and subroutines.

2. The Excel Workbook files may contain multiple worksheets per file. The large master and cross-sectional files were not converted into Excel because of their size. See Section 4.3 for details about the Excel file and work sheet layout.

3. The SAS files are standalone data sets. They should be readable on any supported SAS platform. See Section 4.3 for detail on the SAS File layout.

USE OF CRSP SAMPLE PROGRAMS

Sample programs were developed on an Open VMS system and tested on Sun Solaris, Windows, and Linux Redhat.

Sample programs accompanying the daily bond database are in three categories based on the compilers that must be available to use them.

- Fortran-95 - programs that can access text files only.

- C – programs that can access text files, convert those text files to a more efficient binary form, and read those binary files.

- Fortran-95 via C – Fortran-95 programs that can read converted binary files using underlying C functions.

Recommended Usage:
Recommended usage is to copy programs and data from the `/src` and `/data` folders into a working directory and then compile them with the appropriate make command listed below. The make files will compile subroutines and create all executable programs for the samples. Header files, subroutine files, and conversion programs should not be changed. Main programs and make files can be adapted to support user variations of the sample programs.

FORTRAN-95 Sample Programs:

Compiling:

Windows	`nmake /F f95_daily_bond_samp.mak`
Sun	`make -f f95_daily_bond_samp.mk`
Linux-g95	`make -f f95_daily_bond_samp.mkg5`
Linux-Lahey	`make -f f95_daily_bond_samp.mk5`

Running Programs:

Data files are expected to be in the current working directory. Open statements in the sample programs can be modified to access files in another location.

C Sample Programs:

Compiling:

Windows	`nmake /F c_daily_bond_samp.mak`
Sun	`make -f c_daily_bond_samp.mk`
Linux-Lahey	`make -f c_daily_bond_samp.mk5`

Running Programs:

All programs expect an argument with the path of your data files. All random access programs `*rand.c` expect an additional argument with an input file. Example input files are provided: inpcrspid.dat for `bm*rand.c` programs, and `inpdate.dat` for `bx*rand.c` programs. `bmb*.c` and `bxb*.c` programs require a one-time conversion to binary with `bmc_bmb_conv.c`, which creates the binary Calendar and Master files, or `bxc_bxb_conv.c`, which creates the binary Cross-Sectional files.

For example, on Windows, assuming programs and data are in your working directory, to convert Master files and run a random access program to process selected CRSPIDs, run:

```
bmc_bmb_conv .\
bmb_seq_rand .\ incrspid.dat
```

FORTRAN using C Sample Programs:

Compiling:

- Windows:
  `nmake /F f95_c_daily_bond_samp.mak`
- Sun:
  `make -f f95_c_daily_bond_samp.mk`
- Linux-g95:
  `make -f f95_c_daily_bond_samp.mkg5`
- Linux-Lahey:
  `make -f f95_c_daily_bond_samp.mk5`

Running Programs:

FORTRAN with C programs require a conversion to binary followed by the use of the environment variable, bondpath, described above.

- `bmc_bmb_conv` creates the binary Calendar and Master files.
- `bxc_bxb_conv` creates the binary Cross-Sectional file.

FORTRAN with C requires the user to set a bondpath statement prior to running the executables. The path should be set to the location of the binary data files. Syntax is:

- Windows
  `set bondpath = .\bondpathname\`
- Linux & Sun:
  ```
  bondpath = ./bondpathname/
  export bondpath
  ```

For example, on Windows, after compiling programs, the following steps may be:

```
bmc_bmb_conv .\
set bondpath = .\
bmbfor
```

DESCRIPTION OF PROGRAMS

CRSP provides FORTRAN and C subroutines and sample programs that can be used to access the data in Master or Cross-Sectional File formats. The FORTRAN programs can sequentially read the character files provided and C programs can sequentially or randomly read the character files provided. In addition, there are C programs that can convert the data files to binary and C and FORTRAN programs that can read sequentially or randomly the binary files created.

The following table shows how data items can be accessed in the FORTRAN programs for Master or Cross-Sectional Files. The table is ordered by data item names as described in Section 3. Usage shows whether the data item is being accessed in Master or Cross-Sectional Files. The calendar is available in both groups of files. Common block names are not used when directly accessing a variable in a program.

DATA TYPE	DATA ITEM NAME	FORTRAN DATA TYPE	USAGE	FORTRAN VARIABLE WITH COMMON BLOCK	INDEX I BETWEEN
CALENDAR	QDATE	INTEGER	Calendar	/BXCAL/QDATE[I]	1 and /BXCAL/NQDAT
			Cross-Sectional	/BXCAL/XQDATE	n/a
	DELDAT	INTEGER	Calendar	/BXCAL/DELDAT[I]	1 and /BXCAL/NQDAT
	CD1M	REAL	Calendar	/BXCAL/CD1M[I]	1 and /BXCAL/NQDAT
	CD3M	REAL	Calendar	/BXCAL/CDM3M[I]	1 and /BXCAL/NQDAT
	CD6M	REAL	Calendar	/BXCAL/CD6M[I]	1 and /BXCAL/NQDAT
	CP30D	REAL	Calendar	/BXCAL/CP30D[I]	1 and /BXCAL/NQDAT
	CP60D	REAL	Calendar	/BXCAL/CP60D[I]	1 and /BXCAL/NQDAT
	CP90D	REAL	Calendar	/BXCAL/CP90D[I]	1 and /BXCAL/NQDAT
	FFEFRT	REAL	Calendar	/BXCAL/FFEFRT[I]	1 and /BXCAL/NQDAT
	FFMINR	REAL	Calendar	/BXCAL/FFMINR[I]	1 and /BXCAL/NQDAT
	FFMAXR	REAL	Calendar	/BXCAL/FFMAXR[I]	1 and /BXCAL/NQDAT
	NUMACT	INTEGER	Calendar	/BXCAL/NUMACT [I]	1 and /BXCAL/NQDAT
			Cross-Sectional	/BXHEAD/XNUM	n/a
HEADER	CRSPID	CHARACTER*15	Master	/BMHEAD/CRSPID	n/a
			Cross-Sectional	/BMHEAD/CRSPID[I]	1 and /BXHEAD/XNUM
	TYPE	INTEGER	Master	/BMHEAD/TYPE	n/a
	MATDT	INTEGER	Master	/BMHEAD/MATDT	n/a
	COUPRT	REAL*8	Master	/BMHEAD/COUPRT	n/a
	UNIQ	INTEGER	Master	/BMHEAD/UNIQ	n/a
	WHY	INTEGER	Master	/BMHEAD/WHY	n/a
	DATDT	INTEGER	Master	/BMHEAD/DATDT	n/a
	BANKDT	INTEGER	Master	/BMHEAD/BANKDT	n/a
	FCALDT	INTEGER	Master	/BMHEAD/FCALDT	n/a
	YMCNOT	INTEGER	Master	/BMHEAD/YMCNOT	n/a
	NOTICE	INTEGER	Master	/BMHEAD/NOTICE	n/a
	TAX	INTEGER	Master	/BMHEAD/TAX	n/a
	FLOWER	INTEGER	Master	/BMHEAD/FLOWER	n/a
	NIPPY	INTEGER	Master	/BMHEAD/NIPPY	n/a
	FCPDT	INTEGER	Master	/BMHEAD/FCPDT	n/a
	FCPDTF	INTEGER	Master	/BMHEAD/FCPDTF	n/a
	VALFC	REAL*8	Master	/BMHEAD/VALFC	n/a
	CUSIP	CHARACTER*8	Master	/BMHEAD/CUSIP	n/a
	NAME	CHARACTER*8	Master	/BMHEAD/NAME	n/a
	FSTQUO	INTEGER	Master	/BMHEAD/FSTQUO	n/a
	LSTQUO	INTEGER	Master	/BMHEAD/LSTQUO	n/a
	FSTYLD	INTEGER	Master	/BMHEAD/FSTYLD	n/a
	LSTYLD	INTEGER	Master	/BMHEAD/LSTYLD	n/a
	NUMPAY	INTEGER	Master	/BMHEAD/NUMPAY	n/a
	NUMDBT	INTEGER	Master	/BMHEAD/NUMDBT	n/a

DATA TYPE	DATA ITEM NAME	FORTRAN DATA TYPE	USAGE	FORTRAN VARIABLE WITH COMMON BLOCK	INDEX I BETWEEN
QUOTES	BID	REAL*8	Master	/BMQUO/BID[I]	/BMHEAD/FSTQUO and
			Cross-Sectional	/BXQUO/BID[I]	/BMHEAD/LSTQUO1 and /BXHEAD/XNUM
	ASK	REAL*8	Master	/BMQUO/ASK[I]	/BMHEAD/FSTQUO and /BMHEAD/LSTQUO
			Cross-Sectional	/BXQUO/ASK[I]	/BXQUO/ASK[I] 1 and /BXHEAD/XNUM
	SOURCE	CHARACTER*1	Master	/BMQUO/SOURCE[I]	/BMHEAD/FSTQUO and /BMHEAD/LSTQUO
			Cross-Sectional	/BXQUO/SOURCE[I]	1 and BXHEAD/XNUM
YIELDS	ACCINT	REAL*8	Master	/BMYLD/ACCINT[I]	/BMHEAD/FSTYLD and /BMHEAD/LSTYLD
			Cross-Sectional	/BXYLD/ACCINT[I]	1 and /BXHEAD/XNUM
	YLD	REAL*8	Master	/BMYLD/YLD[I]	/BMHEAD/FSTYLD and /BMHEAD/LSTYLD
			Cross-Sectional	/BXYLD/YLD[I]	1 and /BXHEAD/XNUM
	RETNUA	REAL*8	Master	/BMYLD/RETNUA[I]	/BMHEAD/FSTYLD and /BMHEAD/LSTYLD
			Cross-Sectional	/BXYLD/RETNUA[I]	1 and /BXHEAD/XNUM
	DURATN	REAL*8	Master	/BMYLD/DURATN[I]	/BMHEAD/FSTYLD and /BMHEAD/LSTYLD
			Cross-Sectional	/BXYLD/DURATN[I]	1 and /BXHEAD/XNUM
DEBT	DQDATE	INTEGER	Master	/BMDEBT/DQDATE[I]	1 and /BMHEAD/NUMDBT
	TOTOUT	INTEGER	Master	/BMDEBT/TOTOUT[I]	1 and /BMHEAD/NUMDBT
	PUBOUT	INTEGER	Master	/BMDEBT/PUBOUT[I]	1 and /BMHEAD/NUMDBT
PAYMENTS	PQDATE	INTEGER	Master	/BMPAY/PQDATE[I]	1 and /BMHEAD/NUMPAY
	PDINT	REAL*8	Master	/BMPAY/PDINT[I]	1 and /BMHEAD/NUMPAY

The following table shows how data items can be accessed in the C programs for Master or Cross-Sectional Files. The table is ordered by data item names as described in Section 3. Usage shows whether the data items is being accessed in Master or Cross-Sectional Files. The calendar is available in both groups of files.

DATA TYPE	DATA ITEM NAME	C DATA TYPE	USAGE	C VARIABLE WITH STRUCTURE	INDEX I BETWEEN
CALENDAR	QDATE	int	Calendar	bxcal.qdat [i]	1 and nbx_cal
			Cross-Sectional	bx_struct.bxhead.qdate	n/a
	DELDAT	int	Calendar	bxcal.deldat [i]	1 and nbx_cal
	CD1M	float	Calendar	bxcal.cd1m [i]	1 and nbx_cal
	CD3M	float	Calendar	bxcal.cdm3m [i]	1 and nbx_cal
	CD6M	float	Calendar	bxcal.cd6m [i]	1 and nbx_cal
	CP30D	float	Calendar	bxcal.cp30d [i]	1 and nbx_cal
	CP60D	float	Calendar	bxcal.cp60d [i]	1 and nbx_cal
	CP90D	float	Calendar	bxcal.cp90d [i]	1 and nbx_cal
	FFERT	float	Calendar	bxcal.ffefrt [i]	1 and nbx_cal
	FFMINR	float	Calendar	bxcal.ffminr [i]	1 and nbx_cal
	FFMAXR	float	Calendar	bxcal.ffmaxr [i]	1 and nbx_cal
	NUMACT	int	Calendar	bxcal.numact [i]	1 and nbx_cal
			Cross-Sectional	bx_struct.bxhead.numact	n/a
HEADER	CRSPID	Char[16]	Master	bm_struct.bmhead.crspid	n/a
			Cross-Sectional	bm_struct.bmquo.crspid [i]	0 and <bx_struct.bxhead.numact
			Cross-Sectional	bm_struct.bmyld.crspid [i]	0 and <bx_struct.bxhead.numact
	TYPE	int	Master	bm_struct.bmhead.type	n/a
	MATDT	int	Master	bm_struct.bmhead.matdt	n/a
	COUPRT	double	Master	bm_struct.bmhead.couprt	n/a
	UNIQ	int	Master	bm_struct.bmhead.uniq	n/a
	WHY	int	Master	bm_struct.bmhead.why	n/a
	DATDT	int	Master	bm_struct.bmhead.datdt	n/a
	BANKDT	int	Master	bm_struct.bmhead.bankdt	n/a
	FCALDT	int	Master	bm_struct.bmhead.fcaldt	n/a
	YMCNOT	int	Master	bm_struct.bmhead.ymcnot	n/a
	NOTICE	int	Master	bm_struct.bmhead.notice	n/a
	TAX	int	Master	bm_struct.bmhead.tax	n/a
	FLOWER	int	Master	bm_struct.bmhead.flower	n/a
	FCPDT	int	Master	bm_struct.bmhead.fcpdt	n/a
	FCPDTF	int	Master	bm_struct.bmhead.fcpdtf	n/a
	VALFC	double	Master	bm_struct.bmhead.valfc	n/a
	CUSIP	char[9]	Master	bm_struct.bmhead.cusip	n/a
	NAME	char[9]	Master	bm_struct.bmhead.name	n/a
	FSTQUO	int	Master	bm_struct.bmhead.fstquo	n/a
	LSTQUO	int	Master	bm_struct.bmhead.lstquo	n/a
	FSTYLD	int	Master	bm_struct.bmhead.fstyld	n/a
	LSTYLD	int	Master	bm_struct.bmhead.lstyld	n/a
	NUMPAY	int	Master	bm_struct.bmhead.numpay	n/a
	NUMDBT	int	Master	bm_struct.bmhead.numdbt	n/a

DATA TYPE	DATA ITEM NAME	C DATA TYPE	USAGE	C VARIABLE WITH STRUCTURE	INDEX I BETWEEN
QUOTES	BID	double	Master	bm_struct.bmquo.bid[i]	bm_struct.bmhead.fstquo and bm_struct.bmhead.lstquo
			Cross-Sectional	bx_struct.bxquo.bid [i]	0 and <bx_struct.bxhead.numact
	ASK	double	Master	bm_struct.bmquo.ask [i]	bm_struct.bmhead.fstquo and bm_struct.bmhead.lstquo
			Cross-Sectional	bx_struct.bxquo.ask[i]	0 and <bx_struct.bxhead.numact
	SOURCE	char	Master	bm_struct.bmquo.source [i]	bm_struct.bmhead.fstquo and bm_struct.bmhead.lstquo
			Cross-Sectional	bx_struct.bxquo.source [i]	0 and <bx_struct.bxhead.numact
YIELDS	ACCINT	double	Master	bm_struct.bmyld.accint [i]	bm_struct.bmhead.fstyld and bm_struct.bmhead.lstyld
			Cross-Sectional	bx_struct.bxyld.accint [i]	0 and <bx_struct.bxhead.numact
	YLD	double	Master	bm_struct.bmyld.yld [i]	bm_struct.bmhead.fstyld and bm_struct.bmhead.lstyld
			Cross-Sectional	bx_struct.bxyld.yld[i]	0 and <bx_struct.bxhead.numact
	RETNUA	double	Master	bm_struct.bmyld.retnua [i]	bm_struct.bmhead.fstyld and bm_struct.bmhead.lstyld
			Cross-Sectional	bx_struct.bxyld.retnua [i]	0 and <bx_struct.bxhead.numact
	DURATN	double	Master	bm_struct.bmyld.duratn [i]	bm_struct.bmhead.fstyld and bm_struct.bmhead.lstyld
			Cross-Sectional	bx_struct.bxyld.duratn [i]	0 and <bx_struct.bxhead.numact
DEBT	DQDATE	int	Master	bm_struct.bmdebt.qdate [i]	0 and <bm_struct.bmhead.numdbt
	TOTOUT	int	Master	bm_struct.bmdebt.totout [i]	0 and <bm_struct.bmhead.numdbt
	PUBOUT	int	Master	bm_struct.bmdebt.pubout [i]	0 and <bm_struct.bmhead.numdbt
PAYMENTS	PQDATE	int	Master	bm_struct.bmpay.qdate [i]	0 and <bm_struct.bmhead.numpay
	PDINT	double	Master	bm_struct.bmdebt.pdint [i]	0 and <bm_struct.bmhead.numpay

FORTRAN SAMPLE PROGRAMS

The sample programs give short examples of how to access the data with access routines using FORTRAN. The first two give basic examples of sequential access to the character files using FORTRAN, while the last four illustrate both sequential and random access to the binary files, using C access routines, which are described later in this chapter. To use a sample program, copy it to your directory, edit the program to meet your needs and run according to the instructions inside the program.

CHARACTER FILES

BMSAMP Program BMSAMP reads the character calendar file and the character Master File. BMSAMP first calls subroutine BXCGTC to read the character calendar file into the common block /BXCAL/. BMSAMP then makes successive calls to BMGETC, each call reading all data from the data files for one issue into the common blocks /BMHEAD/ (header information), /BMQUO/ (quotes information), /BMYLD/ (yield information), /BMDEBT/ (debt information) and /BMPAY/ (payment information).

BXSAMP Program BXSAMP reads the character calendar file and the character Cross-Sectional File. BXSAMP first calls subroutine BXCGTC to read the character calendar file into the common block /BXCAL/. BXSAMP then makes successive calls to BXGETC, each call reading all data from the data files for the one quote date into the common blocks /BXHEAD/ (header information), /BXQUO/ (quotes information), /BXYLD/ (yield information).

BINARY FILES

BMBFOR Program BMBFOR reads sequentially the binary Master Files using C access functions. BMBFOR calls subroutine BMBRDK to read a BM_STRUCT structure. It also calls BMBOPE to open the files and load the index, and BMBCLO to close the files.

BMBRAN Program BMBRAN reads randomly the binary Master Files using C access functions. BMBRAN calls subroutine BMBRDK to read a BM_STRUCT structure. It also calls BMBOPE to open the files, and BMBCLO to close the files.

BXBFOR Program BXBFOR reads sequentially the binary Cross-Sectional Files using C access functions. BXBFOR calls subroutine BXBRDK to read a BX_STRUCT structure. It also calls BXBOPE to open the files and load the index, and BXBCLO to close the files.

BXBRAN Program BXBRAN reads sequentially the binary Cross-Sectional Files using C access functions. BXBRAN calls subroutine BXBRDK to read a BX_STRUCT structure. It also calls BXBOPE to open the files and load the index, and BXBCLO to close the files.

FORTRAN ACCESS SUBROUTINES

FORTRAN access subroutines are used by FORTRAN programs to actually retrieve US Treasury data for processing. These subroutines should be included in an object library. You should link the library with each program that uses any of the access functions.

BMGETC (*, *)

Subroutine BMGETC first calls BMRES to erase the previous record's data, and then it reads all data from teh data files for one issue into the common blocks /BMHEAD/ (header information), /BXQUO/ (quotes information), /BMYLD/ (yield information), /BMDEBT/ (debt information) and /BMPAY/ (payment information). BMGETC first reads a header record and then reads LSTQUO - FSTQUO + 1 quotes records, LSTYLD - FSTYLD + 1 yield records, NUMDBT debt records and NUMPAY payment records. BMGETC makes sure that the CRSPID from the header and the data records are the same. The first alternate return is taken from the file. The second alternate return is only taken if there is an error.

BXGETC (THEDAY, NUMREC, *,*)

Subroutine BXGETC first calls BXRES to erase the previous record's data and then reads all data for one quote date from the data files into the common blocks /BXHEAD/ (header information), /BXYLD/ (yield information). BXGETC has two parameters:

THEDAY - the quote date

NUMREC - the number of issues having the THEDAY quote date

BXGETC reads NUMREC quotes records and then NUMREC yield records. BXGETC makes sure that the parameter THEDAY and the quote date of the data records are the same and that the CRSPID of the quotes data is the same as the CRSPID of the yield data. The first alternate return is taken at the end of the file. The second alternate return is only taken if there is an error.

BXCGTC Subroutine BXCGTC reads the character calendar file into the /BXCAL/ common block.

BXCGTC reads the variables QDATE, DELDAT, various rates, and NUMACT into the /BXCAL/ common block from the character calendar file. It assumes that the file BXCALIND.DAT is opened and the unit number is IUNIT6. NQDAT is set to the number of days read from the calendar file.

FORTRAN UTILITY SUBROUTINES

FORTRAN utility subroutines are used by FORTRAN programs to actually obtain different CRSP derived variables. These subroutines should also be included into the object library. You should link the library with each program that uses any of the utility functions.

SUBROUTINE	TYPE	DESCRIPTION
BMRES	BM	reset master structure
BXRES	BX	reset cross-sectional structure
BXCLJL	CAL	convert calendar date to Julian date
FPDINT	BM	derive paid interest for a date
IDBT	CAL	find index in debt array for a date
INDCAL	CAL	find index in a calendar for a date
INDCID	BX	find index in a CRSPID list for a CRSPID
IPAY	BM	find index in payment structure for a date
IQDAY	CAL	find DD day for a calendar index
JAHRMO	CAL	find year and month for a calendar index
NDDATE	CAL	find Julian day number of delivery date for a calendar index
NDHFYR	CAL	return number of days in last half year
NDIFDT	CAL	find difference in days between 2 dates
NDZERO	CAL	find zero'th day of a month
NPOUT	BM	find publicly held value for calendar index
NQDATE	CAL	Julian day number for calendar index
NQTOQD	CAL	find number of days between given index and previous
NTOUT	BM	find total debt for calendar index
PCYIELD	BM	calculate yield to maturity compounded to given frequency
RETADJ	BM, BX	express holding period return as a percentage
YTM	BM, BX	calculate annualized yield to maturity

BMRES Subroutine BMRES resets the vectors belonging to the previous master structure. It initializes the /BMQUO/, /BMYLD/, /BMDEBT/, and /BMPAY/ common blocks.

BXRES Subroutine BXRES resets the vectors belonging to the previous master structure. It initializes the /BXHEAD/, /BMQUO/, and /BMYLD/ common blocks.

BXCLJL (IDTCAL, IDTJUL, *)

Subroutine BXCLJL converts a

calendar date to its linear (Julian) date equivalent. IDTCAL is the integer YYYYMMDD date which BXCLJL should convert, IDTJUL is the converted (Julian) date which BXCLJL returns. The alternative return is used if IDTCAL is an illegal date.

INTEGER FPDINT (IDXCAL)

Function FPDINT takes as a parameter IDXCAL - index in the calendar, calls the IPAY function to get the index in the BMPAY vector corresponding to the calendar data and returns the paid interest for that date. FPDINT returns -1 if the date was not found.

INTEGER IDBT (IDXCAL)

Function IDBT takes as a parameter IDXCAL - index in the calendar, searches in the BMDEBT vector and returns the index in the BMDEBT vector corresponding to the calendar data. IDBT returns -1 if the date was not found.

INTEGER INDCAL (DATE, CODE, ARRAY, MAXARR)

Function INDCAL can be used to locate the index of a date in a given date array. DATE is the value to be located in array ARRAY with MAXARR sorted values. CODE is either -1, 0, 1, depending of what action is taken when the exact given date is not found. If CODE = 0 and the exact date is not found, 0 is returned. If CODE = -1 and the exact date is not found, the index of the first date less than DATE is returned, or 0 is returned if DATE is less than any date in the array. If CODE = 1 and the exact date is not found, the index of the first date greater than DATE will be returned, or 0 is returned if DATE is greater than any date in the array.

INTEGER INDCID (CRSPID, CODE, ARRAY, MAXARR)

Function INDCID can be used to locate the index of a CRSPID in a given CRSPIDs array. CRSPID is the value to be located in array ARRAY with MAXARR sorted values. CODE is either -1, 0, 1, depending of what action is taken when the CRSPID is not found. If CODE = 0 and the CRSPID is not found, 0 is returned. If CODE = -1 and the CRSPID is not found, the index of the previous CRSPID in the array is returned, or 0 is returned if CRSPID is the first one in the array. If CODE = 1 and the CRSPID is not found, the index of the next CRSPID in the array will be returned, or 0 is returned if CRSPID is the last one in the array.

INTEGER IPAY (IDXCAL)

Function IPAY takes as a parameter IDXCAL - index in the calendar, searches in the BMPAY vector and returns the index in the BMPAY vector corresponding to the calendar data. IPAY returns -1 if the date was not found.

INTEGER IQDAY (IDXCAL)

Function IQDAY takes as a parameter IDXCAL and returns the day (DD) of the quotation date which has index IDXCAL. Returns -1 if IDXCAL is out of range.

INTEGER JAHRMO (IDXCAL)

Function JAHRMO takes as a parameter IDXCAL and returns the year and month (YYYYMM) of the quotation date which has index IDXCAL. Returns -1 if IDXCAL is out of range.

INTEGER NDDATE (IDXCAL)

Function NDDATE takes as a parameter IDXCAL and returns the number of days of the delivery date which have index IDXCAL. NDDATE calls the BXCLJL function to get the day number. Returns -1 if IDXCAL is out of range or if BXCLJL fails.

INTEGER NDHFYR (IDXCAL)

Function NDHFYR takes as a parameter IDXCAL and returns the number of days in the last half year corresponding to the quotation date which has index IDXCAL. NDHFYR calls the NDIFDT function to get the difference between the quotation date. Returns -1 if IDXCAL is out of range.

INTEGER NDIFDT (IDAT1, IDAT2)

Function NDIFDT converts two calendar dates to linear (Julian) dates and returns the difference. IDAT1 and IDAT2 are integer YYYYMMDD dates. NDIFDT calls the BXCLJL function to calculate the linear (Julian) dates.

INGETER NDZERO (IDXCAL)

Function NDZERO takes as a parameter IDXCAL and returns the zero'th day of the month of the quotation date which has index IDXCAL. NQDATE calls the BXCLJL function to get the linear date. Returns -1 if IDXCAL is out of range or if BXCLJL fails.

INTEGER NPOUT (IDXCAL)

Function NPOUT takes as a parameter IDXCAL - index in the calendar, calls the IDBT function to get the index in the BMDEBT vector corresponding to the calendar data and returns the publicly held face value outstanding for that date. NPOUT returns -1 if the date was not found.

INTEGER NQDATE (IDXCAL)

Function NQDATE takes as a parameter IDXCAL and returns the day number of the quotation date which has index IDXCAL. NQDATE calls the BXCLJL function to get the day number. Returns -1 if IDXCAL is out of range or if BXCLJL fails.

INTEGER NQTOQD (IDXCAL)

Function NQTOQD takes as a parameter IDXCAL and returns the number of days between the previous quotation date and the quotation date which has index IDXCAL. NQTOQD calls the NQDATE function to get the linear (Julian) quotation dates. Returns -1 if IDXCAL is out of range.

INTEGER NTOUT (IDXCAL)

Function NTOUT takes as a parameter IDXCAL - index in the calendar, calls the IDBT function to get the index in the BMDEBT vector corresponding to the calendar data and returns the face value outstanding for that date. NTOUT returns -1 if the date was not found.

PCYLD (PCYARR, FREQ)

Subroutine PCYLD calculates the yield to maturity. PCYLD has two parameters:

PCYARR - an array of floats which will be loaded with the calculated values.

FREQ - the frequency.

If a yield is missing, the value will be -99.

RETADJ (ADJARR)

Subroutine RETADJ calculates the holding period return expressed as a percentage. RETADJ has a parameter:

ADJARR - an array of floats which will be loaded with the calculated values.

If RETNUA, the unadjusted return, is missing, the value will be -999.

YTM (YTMARR)

Subroutine YTM calculates the annualized yield to maturity. YTM has a parameter:

YTMARR - an array of floats which will be loaded with the calculated values.

If a yield is missing, the value will be -999.

FORTRAN INCLUDE FILES

The sample programs and subroutines use include files to replace long, often-used blocks of code with single statements. If an include file is modified, all programs and subroutines that use the include file must be recompiled. All declarations needed to use the CRSP data with FORTRAN programs are automatically made by adding the include statements at the beginning of any main programs or subprograms that will use CRSP data or CRSP access or utility routines.

BMINCL Include file BMINCL contains constants definitions and common blocks definitions to be used in any program or subroutine which access the Master Files.

BXINCL Include file BXINCL contains constants definitions and common blocks definitions to be used in any program or subroutine which access the Cross-Sectional Files.

CALINC Include file CALINC contains constants definitions and common blocks definitions to be used in any program or subroutine which access the Calendar File.

BMBPRM Include file BMBPRM contains constants definitions to be used by programs or subroutines which access the Master Files using C functions.

BXBPRM Include file BXBPRM contains constants definition to be used by programs or subroutines which access the Cross-Sectional Files using C functions.

C SAMPLE PROGRAMS

The sample programs give short examples of how to access the data with the access routines using C. The conversion programs generate the binary files from the character files. The character programs give basic examples of the C sequential and random access to the character files, and the binary programs illustrate both sequential and random access to the binary files. To use a sample program, copy it to your directory, edit the program to meet your needs and run according to the instructions inside the program.

Example of Usage:

```
bmc_bmb_conv bndpath
```

CONVERSION PROGRAMS FROM CHARACTER TO BINARY

ACCESS ROUTINE:	BMC_BMB_CONV
Description:	Reads the character calendar file and then reads sequentially the character files and writes into binary files the loaded structure.
Methodology:	bmc_bmb_conv first calls procedure bxc_cal_load to load the calendar in the bx_cal array, reads bond data one CRSPID at a time until the end of files and then writes the data into the binary files. The function bmc_rdkey loads all wanted data into the bms structure for the next CRSPID and the function bmb_wrkey writes the structure into the binary files. The program also calls the bxb_cal_write to write the calendar array into the binary calendar file.
Parameters:	bndpath -The path of the data files must be the parameter set on the command line. This is the location of the text data files and will be the location of the new binary files created by the conversion program.
Return Values:	
Notes:	Converts the Master Files from character to binary.

ACCESS ROUTINE:	BXC_BXB_CONV
Description:	Reads sequentially the character Cross-Sectional Files and writes the data into the binary Cross-Sectional Files.
Methodology:	bxc_bxb_conv reads character data on each date until the end of files, then writes it into the binary files. The function bxc_rdkey loads all wanted data into the bxs structure for the next date and the function bxb_wrkey write the structure into the files.
Parameters:	bndpath - The path of the data files must be the parameter set on the command line. This is the location of the text data files and will be the location of the new binary files created by the conversion program.
Return Values:	
Notes:	Converts the Cross-Sectional Files from character to binary.

CHARACTER FILES

ACCESS ROUTINE:	BMC_READ_RAND
Description:	reads the character calendar file and then reads randomly the character Master Files.
Sample Usage:	bmc_read_rand bndpath inpfilename
Methodology:	bmc_read_rand first calls procedure bxc_cal_load to load the calendar in the bx_cal array and then reads sequentially the input file and calls the bxc_rdkey for each read CRSPID. The function bxc_rdkey loads all wanted data into the bms structure for the desired CRSPID.
Parameters:	bndpath - the path of the directory where the files are.
	inpfilename - the input file name (including the path). inpcrspid.dat
Return Values:	
Notes:	The wanted CRSPIDs are read from a text file.

ACCESS ROUTINE:	BMC_READ_SEQ
Description:	Reads the character calendar file and then reads sequentially the character Master Files.
Methodology:	bmc_read_seq first calls procedure bxc_cal_load to load the calendar in the bx_cal array and then reads data one CRSPID by one till the end of files. The function bmc_rdkey loads all wanted data into the bms structure for the next CRSPID.
Parameters:	bndpath - the path of the directory where the files are.
Return Values:	
Notes:	

ACCESS ROUTINE:	BXC_READ_RAND
Description:	Reads randomly the character Cross-Sectional Files.
Sample Usage:	bxc_read_rand bndpath inpfilename

ACCESS ROUTINE:	BXC_READ_RAND
Methodology:	bxc_read_rand reads sequentially the input file and calls the bxc_rdkey for each read date. The function bxc_rdkey loads all wanted data into the bxs structure for the desired date.
Parameters:	bndpath - the path of the directory where the files are.
	inpfilename - the input file name (including the path). inpdate.dat
Return Values:	
Notes:	The wanted dates are read from a text file.

Access Routine:	bxc_read_seq
Description:	Reads sequentially the character Cross-Sectional Files.
Methodology:	bxc_read_seq reads data in a loop till the end of files. The function bxc_rdkey loads all wanted data into the bxs structure for the next date.
Parameters:	bndpath - the path of the directory where the files are.
Return Values:	
Notes:	

BINARY FILES

ACCESS ROUTINE:	BMB_READ_RAND
Description:	Reads the binary calendar file and then reads randomly the binary Master Files.
Sample Usage:	bmb_read_rand bndpath inpfilename
Methodology:	bmb_read_rand first calls procedure bxb_cal_load to load the calendar in the bx_cal array and then reads sequentially the input file and calls the bxb_rdkey for each read CRSPID. The function bxb_rdkey loads all wanted data into the bms structure for the desired CRSPID.
Parameters:	bndpath - the path of the directory where the files are.
	inpfilename - the input file name(including the path). inpcrspid.dat
Return Values:	
Notes:	The wanted CRSPIDs are read from a text file.

ACCESS ROUTINE:	BMB_READ_SEQ
Description:	Reads the binary calendar file and then reads sequentially the binary Master Files.
Methodology:	bmb_read_seq first calls procedure bxb_cal_load to load the calendar in the bx_cal array and then reads bond data one CRSPID by one till the end of files. The function bmb_rdkey loads all wanted data into the bms structure for the next CRSPID.
Parameters:	bndpath - the path of the directory where the files are.
Return Values:	

ACCESS ROUTINE:	BXB_READ_RAND
Description:	Reads randomly the binary Cross-Sectional Files.
Sample Usage:	bxb_read_rand bndpath inpfilename
Methodology:	bxb_read_rand reads sequentially the input file and calls the bxb_rdkey for each read date. The function bxb_rdkey loads all wanted data into the bxs structure for the desired date.
Parameters:	bndpath - the path of the directory where the files are.
	inpfilename - the input file name(including the path). inpdate.dat
Return Values:	
Notes:	The wanted dates are read from a text file.

ACCESS ROUTINE:	BXB_READ_SEQ
Description:	Reads sequentially the binary Cross-Sectional Files.
Methodology:	bxb_read_seq reads bond data in a loop till the end of files. The function bxb_rdkey loads all wanted data into the bxs structure for the next date.
Parameters:	bndpath - the path of the directory where the files are.
Return Values:	
Notes:	

C ACCESS ROUTINES

FUNCTIONS CALLED BY C PROGRAMS

C access subroutines are used by C programs to actually retrieve daily US Treasury data for processing. These subroutines should be included into an object library. Link the library with each program that uses any of the access functions.

CHARACTER FILES

FUNCTION:	BMC_RDKEY
Prototype:	int bmc_rdkey (bm_str, key, wanted)
Description:	Reads the data from the character Master Files for the given key.
Parameters:	bm_str - pointer to the BM_STRUCT structure to be loaded.
	key - the desired CRSPID for random access or MFIRST, MPREV, MLAST, MSAME, MNEXT.
	wanted - the desired information; should be QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM or any combination.
Return Values:	success (0) the key found
	error (-1) for:
	-no read access
	-key not found or no previous for same
	-could not read a needed file
	EOFL (-2)

FUNCTION:	BXC_RDKEY
Prototype:	int bxc_rdkey (bx_str, key, wanted)
Description:	Reads the data from the character Cross-Sectional Files for the given key.
Parameters:	bx_str - pointer to the BX_STRUCT structure to be loaded.
	key - the desired qdate for random access or XFIRST, XPREV, XLAST, XSAME, XNEXT.
	wanted - the desired information; should be QUOTES, YIELDS, ALLBX or any combination.
Return Values:	success (0) the key found
	error (-1) for:
	-no read access
	-key not found or no previous for same
	-could not read a needed file
	EOFL (-2)

FUNCTION:	BXC_CAL_LOAD
Prototype:	int bxc_cal_load (bndpath)
Description:	Function to load the character calendar from the BXCALIND.DAT file into the bx_cal array.
Parameters:	bndpath - the path of the directory where the calendar file is.
Return Values:	success(nbx_cal) - the number of dates
	error (-1)

FUNCTION:	BMC_OPEN
Prototype:	int bmc_open (bndpath,wanted)
Description:	Opens wanted character Master Files and loads the index file in an array.
Parameters:	bndpath - the path of the directory where the data files are located.
	wanted - the desired information should be QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM or any combination.
Return Values:	success (0)
	error (-1)

FUNCTION:	BXC_OPEN
Prototype:	int bxc_open (bndpath,wanted)

FUNCTION:	BXC_OPEN
Description:	Opens wanted character Cross-Sectional Files and loads the index file in an array.
Parameters:	`bndpath` - the path of the directory where the data files are located.
	`wanted` - the desired information should be `QUOTES, YIELDS, ALLBX` or any combination.
Return Values:	success (0)
	error (-1)

FUNCTION:	BMC_CLOSE
Prototype:	`int bmc_close (wanted)`
Description:	Opens wanted character in Master Files sequentially.
Parameters:	wanted - the desired information should be `QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM` or any combination.
Return Values:	success (0)
	couldn't close (-1)

FUNCTION:	BXC_CLOSE
Prototype:	`int bxc_close (wanted)`
Description:	Opens wanted character in Cross-Sectional Files sequentially.
Parameters:	wanted - the desired information should be `QUOTES, YIELDS, ALLBX,` or any combination.
Return Values:	success (0)
	couldn't close(-1)

BINARY FILES

FUNCTION:	BMB_RDKEY
Prototype:	`int bmb_rdkey (bm_str, key, wanted)`
Description:	Reads the data from the binary Master Files for the given key.
Parameters:	`bm_str` - pointer to the `BM_STRUCT` structure to be loaded.
	`key` - the desired `CRSPID` for random access or `MFIRST, MPREV, MLAST, MSAME, MNEXT.`
	`wanted` - the desired information; should be `QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM` or any combination.
Return Values:	success (0)
	error (-1) for:
	-no read access -key not found or no previous for same -could not read a needed file
	`EOFL` (-2)

FUNCTION:	BXB_RDKEY
Prototype:	`int bxb_rdkey (bx_str, key, wanted)`
Description:	Reads the data from the binary Cross-Sectional Files for the given key.
Parameters:	`bx_str` - pointer to the `BX_STRUCT` structure to be loaded.
	`key` - the desired `CRSPID` for random access or `XFIRST, XPREV, MLAST, MSAME, MNEXT.`
	`Wanted` - the desired information; should be `QUOTES, YIELDS, ALLBX` or any combination.
Return Values:	success (0)
	error (-1) for:
	-no read access -key not found or no previous for same -could not read a needed file
	`EOFL` (-2)

FUNCTION:	BMB_WRKEY
Prototype:	`int bmb_wrkey (bm_str)`
Description:	Writes the data from the `bm_str` structure into the binary Master Files.
Parameters:	`bm_str` - pointer to the `BM_STRUCT` structure to be loaded.
Return Values:	success (0) - the key found
	error (-1)

FUNCTION:	BXB_WRKEY
Prototype:	`int bxb_wrkey (bx_str)`
Description:	Writes the data from `bx_str` structure into the binary Cross-Sectional Files.
Parameters:	`bx_str` - pointer to the `BX_STRUCT` structure to be loaded.
Return Values:	success (0)
	error (-1)

FUNCTION:	BXB_CAL_LOAD
Prototype:	`int bxb_cal_load (bndpath)`
Description:	Function to load the binary calendar from the `BXCALIND.DAT` file into the `bx_cal` array.
Parameters:	`bndpath` - the path of the directory where the calendar file is.
Return Values:	success (`nbx_cal`) - the number of dates
	error (-1)

FUNCTION:	BMB_OPEN
Prototype:	`int bmb_open (bndpath,wanted)`
Description:	Opens wanted binary Master Files and loads the index file in an array.

FUNCTION:	BMB_OPEN
Parameters:	bndpath - the path of the directory where the data files are.
	wanted - the desired information; should be QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM or any combination.
Return Values:	success (0)
	error (-1)

FUNCTION:	BMB_CLOSE
Prototype:	int bmb_close (wanted)
Description:	Close wanted binary Master Files sequentially.
Parameters:	wanted - the desired information; should be QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM or any combination.
Return Values:	success (0)
	error (-1) - couldn't close

FUNCTION:	BXB_OPEN
Prototype:	int bxb_open (bndpath,wanted)
Description:	Opens wanted binary Cross-Sectional Files and loads the index file in an array.
Parameters:	bndpath - the path of the directory where the data files are.
	wanted - the desired information; should be QUOTES, YIELDS, ALLBX or any combination.
Return Values:	success (0)
	error (-1) - couldn't close

FUNCTION:	BXB_CLOSE
Prototype:	int bxb_close (wanted)
Description:	Close wanted binary Cross-Sectional Files sequentially.
Parameters:	wanted - the desired information; should be QUOTES, YIELDS, ALLBX or any combination.
Return Values:	success (0)
	error (-1) - couldn't close

FUNCTIONS CALLED BY FORTRAN PROGRAMS

FUNCTION:	BMBRDK
Prototype:	int bmbrdk (pbmhead, pbmquo, pbmyld, pbmpay, pbmdebt, key, wanted, ret)
Description:	Reads the data from the master binary files for a given key and loads them into a BM_STRUCT structure and then into FORTRAN common blocks to be used by FORTRAN programs.
Parameters:	pbmhead - pointer to the /BMHEAD/ common block.
	pbmquo - pointer to the /BMQUO/ common block.
	pbmyld - pointer to the /BMYLD/ common block.
	pbmpay - pointer to the /BMPAY/ common block.
	pbmdebt - pointer to the /BMDEBT/ common block.
	key - the desired CRSPID for random access or MFIRST, MPREV, MLAST, MSAME, MNEXT.
	wanted - the desired information; should be QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM or any combination.
	ret - the return code.
Return Values:	success (0) the key found, or
	error (-1) for:
	-no read access -key not found or no previous for same -could not read a needed file
	EOFL (-2)

FUNCTION:	BXBRDK
Prototype:	int bxbrdk (pbxhead, pbxquo, pbxyld, key, wanted, ret)
Description:	Reads the data from the cross-sectional binary files for a given key and load them into a BX_STRUCT structure and then into FORTRAN common blocks to be used by FORTRAN programs.
Parameters:	pbxhead - pointer to the /BXHEAD/ common block.
	pbxquo - pointer to the /BXQUO/ common block.
	pbxyld - pointer to the /BXYLD/ common block.
	key - the desired CRSPID for random access or XFIRST, XPREV, XLAST, XSAME, XNEXT.
	wanted - the desired information; should be QUOTES, YIELDS, ALLBX or any combination.
	ret - the return code.
Return Values:	success (0) the key found, or
	error (-1) for:
	-no read access -key not found or no previous for same -could not read a needed file
	EOFL (-2)

FUNCTION:	BXBCAL
Prototype:	`int bxbcal (pbxcal, nbxcal, bndpath, bndlen, ret)`
Description:	Reads the data from the binary calendar file and load them into FORTRAN common block to be used by FORTRAN programs.
Parameters:	`pbxcal` - pointer to the /`BXCAL`/ common block.
	`nbxcal` - the number of dates.
	`bndpath` - the path of the directory where the file is.
	`bndlen` - the length of the path.
	`ret` - the return code.
Return Values:	success (0) the key found, or
	error (-1)

FUNCTION:	BMBOPE
Prototype:	`int bmbope (bndpath, bndlen, wanted, mode, ret)`
Description:	Opens all master binary data files and loads the index file in the array. It calls the C function `bmb_open`.
Parameters:	`bndpath` - the path of the directory where the data files are.
	`bndlen` - the length of the path.
	`wanted` - the desired information; should be `QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM` or any combination.
	`mode` - the mode to open the files should be “R” (read) or “W” (write).
	`ret` - the return code.
Return Values:	success (0) the key found, or
	error (-1)

FUNCTION:	BMBCLO
Prototype:	`int bmbclo (wanted, ret)`
Description:	Closes the master binary data files sequentially.
Parameters:	`wanted` - the desired information; should be `QUOTES, YIELDS, PAYMTS, DEBTS, ALLBM` or any combination.
	`ret` - the return code.
Return Values:	success (0), or
	error (-1)

FUNCTION:	BXBOPE
Prototype:	`int bxbope (bndpath, bndlen, wanted, mode, ret)`
Description:	Opens all cross-sectional binary data files and loads the index file in the array calling the C function `bxb_open`.

FUNCTION:	BXBOPE
Parameters:	`bndpath` - the path of the directory where the data files are.
	`bndlen` - the length of the path.
	`wanted` - the desired information; should be `QUOTES, YIELDS, ALLBX` or any combination.
	`mode` - the mode to open the files should be “R” (read) or “W” (write).
	`ret` - the return code.
Return Values:	success (0), or
	error (-1)

FUNCTION:	BXBCLO
Prototype:	`int bxbclo (wanted, ret)`
Description:	Close cross-sectional binary data files sequentially.
Parameters:	`wanted` - the desired information; should be `QUOTES, YIELDS, ALLBX` or any combination.
	`ret` - the return code.
Return Values:	success (0), or
	error (-1)

C UTILITY ROUTINES

C utility subroutines are used by C programs to actually obtain different CRSP derived variables. These subroutines should also be included in the object library. Link the library with each program that uses any of the utility functions.

SUBROUTINE	TYPE	DESCRIPTION
`bxcljl`	`cal`	convert calendar date to Julian date
`fpdint`	`bm`	derive paid interest for a date
`idbt`	`cal`	find index in debt array for a date
`indcal`	`cal`	find index in a calendar for a date
`indcid`	`bx`	find index in a `CRSPID` list for a `CRSPID`
`ipay`	`bm`	find index in payment structure for a date
`iqday`	`cal`	find `DD` day for a calendar index
`jahrmo`	`cal`	find year and month for a calendar index
`nddate`	`cal`	find Julian day number of delivery date for a calendar index
`ndhfyr`	`cal`	return number of days in last half year
`ndifdt`	`cal`	find difference in days between 2 dates
`ndzero`	`cal`	find zero’th day of a month
`npout`	`bm`	find publicly held value for calendar index
`nqtoqd`	`cal`	find number of days between given index and previous
`ntout`	`bm`	find total debt for calendar index

SUBROUTINE	TYPE	DESCRIPTION
pcyield	bm	calculate yield to maturity compounded to given frequency
retadj	bm, bx	express holding period return as a percentage
ytm	bm, bx	calculate annualized yield to maturity

UTILITY:	BXCLJL
Prototype:	int bxcljl (idtcal)
Description:	Function bxcljl converts a calendar date to its linear (Julian) date equivalent.
Parameters:	idtcal - date in format YYYYMMDD.
Return Values:	success: the linear date
	error (-1)

UTILITY:	FPDINT
Prototype:	int fpdint (bm_str, idxcal)
Description:	Function fpdint takes as a parameter idxcal - index in the calendar, calls the ipay function to get the index in the bmpay vector corresponding to the calendar data and returns the paid interest for that date.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	idxcal - the index in the calendar.
Return Values:	success: an index in the BMPAY structure
	error (-1)
	fpdint returns -1 if the date was not found.

UTILITY:	IDBT
Prototype:	int idbt(bm_str, idxcal)
Description:	Function idbt takes as a parameter idxcal - index in the calendar, searches in the bmdebt vector and returns the index in the bmdebt vector corresponding to the calendar data.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	idxcal - the index in the calendar.
Return Values:	success: an index in the bmdebt structure
	error (-1)

UTILITY:	INDCAL
Prototype:	int indcal (key, code, array, maxarr)
Description:	indcal can be used to locate the index of a YYYYMMDD date in a calendar array.
Parameters:	key - pointer to a string containing a YYYYMMDD calendar date to find.
	code -1, 0, 1 for handling non-exact matches.
	-1, if date is not found, returns index of previous valid date.
	0, if date is not found, returns 0.
	1, if date is not found, returns index of next valid date.
	array - pointer to an array of YYYYMMDD calendar dates.
	maxarr - number of calendar dates in array.
Return Values:	success: index in array of YYYYMMDD calendar dates
	error(0) if not found or out of range according to code

UTILITY:	INDCID
Prototype:	int indcid (key, code, array, maxarr)
Description:	indcid can be used to locate the index of a CRSPID in a CRSPID array.
Parameters:	key - pointer to a string containing a CRSPID to find.
	code -1, 0, 1 for handling non-exact matches.
	-1, if CRSPID is not found, returns index of previous CRSPID.
	0, if CRSPID is not found, returns 0.
	1, if CRSPID is not found, returns index of next CRSPID.
	array - pointer to an array of CRSPIDs.
	maxarr - number of CRSPIDs in array.
Return Values:	success: index in array of CRSPIDs
	error(0) if not found or out of range according to code

UTILITY:	IPAY
Prototype:	int ipay (bm_str, idxcal)
Description:	Function ipay takes as a parameter idxcal - index in the calendar, searches in the bmpay vector and returns the index in the bmpay vector corresponding to the calendar data.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	idxcal - the index in the calendar.
Return Values:	success: an index in the bmpay structure
	error (-1)

UTILITY:	IQDAY
Prototype:	int iqday (idxcal)
Description:	Function iqday takes as a parameter idxcal and returns the day (DD) of the quotation date which has index idxcal.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: the day of the quotation date
	error (-1)

UTILITY:	JAHRMO
Prototype:	int jahrmo (idxcal)
Description:	Function jahrmo takes as a parameter idxcal and returns the year and month (YYYYMM) of the quotation date which has index idxcal.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: year and month of the quote date YYYYMM
	error (-1)

UTILITY:	NDDATE
Prototype:	int nddate (idxcal)
Description:	Function nddate takes as a parameter idxcal and returns the day number of the of the delivery date which has index idxcal. nddate calls the bxcljl function to get the day number.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: the day number of the delivery date
	error (-1)

UTILITY:	NDHFYR
Prototype:	int ndhfyr (idxcal)
Description:	Function ndhfyr takes as a parameter idxcal and returns the number of days in the last half year corresponding to the quotation date which has index idxcal. ndhfyr calls the ndifdt function to get the difference between the quotation date.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: the linear number of dates in half year
	error (-1)

UTILITY:	NDIFDT
Prototype:	int ndifdt (idat1, idat2)
Description:	Function ndifdt converts two calendar dates to linear (Julian) dates and returns the difference.
Parameters:	idat1 - first date in format YYYYMMDD.
	idat2 - second date in format YYYYMMDD.
Return Values:	success: the difference between idat1 and idat2
	error (-1)
	ndifdt calls the BXCLJL function to calculate the linear (Julian) dates.

UTILITY:	NDZERO
Prototype:	int ndzero (idxcal)
Description:	Function ndzero takes as a parameter idxcal and returns the zero'th day of the month of the quotation date which has index idxcal. nqdate calls the bxcljl function to get the linear date.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: the day number of the zero'th day of the month
	error (-1)

UTILITY:	NPOUT
Prototype:	int npout (bm_str, idxcal)
Description:	Function npout takes as a parameter idxcal - index in the calendar, calls the idbt function to get the index in the bmdebt vector corresponding to the calendar data and returns the publicly held face value outstanding for that date. npout returns -1 if the date was not found.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	idxcal - the index in the calendar.
Return Values:	success: an index in the bmdebt structure
	error (-1)

UTILITY:	NQDATE
Prototype:	int nqdate (idxcal)
Description:	Function nqdate takes as a parameter idxcal and returns the day number of the quotation date which has index idxcal. nqdate calls the bxcljl function to get the day number.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: the day number of the quotation date
	error (-1)

UTILITY:	NQTOQD
Prototype:	int nqtoqd (idxcal)
Description:	Function nqtoqd takes as a parameter idxcal and returns the number of days between the previous quotation date and the quotation date which has index idxcal. nqtoqd calls the nqdate function to get the linear (Julian) quotation dates.
Parameters:	idxcal - the index in the calendar.
Return Values:	success: the number of days between the last the quotation date and this quotation date
	error (-1)

UTILITY:	NTOUT
Prototype:	int ntout (bm_str, idxcal)
Description:	Function ntout takes as a parameter idxcal - index in the calendar, calls the idbt function to get the index in the bmdebt vector corresponding to the calendar data and returns the face value outstanding for that date. ntout returns -1 if the date was not found.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.

UTILITY:	NTOUT
	idxcal - the index in the calendar.
Return Values:	success: an index in the BMDEBT structure
	error (-1)

UTILITY:	PCYIELD
Prototype:	void pcyield (bm_str, pcyarr, freq)
Description:	pcyield calculates the yield to maturity and loads the pcyarr with the results.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	pcyarr - pointer to an array of floats.
	freq - the frequency.
Return Values:	success: none
	error: if a yield is missing, the value will be -99

UTILITY:	RETADJ
Prototype:	void retadj (bm_str, adjarr, freq)
Description:	retadj calculates the holding period return expressed as a percentage and loads the adjarr with the results.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	adjarr - pointer to an array of floats.
Return Values:	success: none
	error: If RETNUA is missing, the value will be -999

UTILITY:	YTM
Prototype:	void ytm (bm_str, ytmarr, freq)
Description:	ytm calculates the annualized yield to maturity and loads the ytmarr with the results.
Parameters:	bm_str - pointer to a BM_STRUCT structure which must be loaded before this function to be called.
	ytmarr - pointer to an array of floats.
Return Values:	success: none
	error: If a yield Is missing, the value will be -999

C INPUT/OUTPUT ROUTINES

These functions are specific to file access (open, close, read sequentially, read randomly). They should be modified according to the compiler's requirements.

ROUTINE:	FILE_OPEN
Prototype:	int file_open (filepath, filename)
Description:	opens a file.
Parameters:	filepath - the path of the directory where the file is.
	filename - the name of the file.

ROUTINE:	FILE_OPEN
Return Values:	success the file descriptor(>0)
	error (-1)

ROUTINE:	FILE_READ
Prototype:	int file_read (fdes, buffer, offset, size)
Description:	reads and stores the data temporary in a character buffer.
Parameters:	fdes - file descriptor.
	buffer - character buffer where the data is read.
	offset - the offset from the beginning of the file from where to read.
	size - the number of bytes to be read.
Return Values:	success (0)
	error (-1)

ROUTINE:	FILE_WRITE
Prototype:	int file_write (fdes, buffer, size)
Description:	Writes a buffer into a file.
Parameters:	fdes - file descriptor.
	buffer - the address of the buffer.
	size - the number of bytes to be read.
Return Values:	success (0)
	error (-1)

ROUTINE:	FILE_NEXT
Prototype:	int file_next (fdes, buffer, size)
Description:	Reads sequentially the next record and stores the data temporary in a character buffer.
Parameters:	fdes - file descriptor.
	buffer - character buffer where the data is read.
	size - the number of bytes to be read.
Return Values:	Success(the number of bytes)
	EOF or error (-1)

ROUTINE:	FILE_CLOSE
Prototype:	int file_close(fdes)
Description:	Close a file.
Parameters:	fdes - file descriptor.
Return Values:	success (0)
	error (-1)

C INCLUDE FILES

The following include files were used in the sample programs, function and procedures. If an include file is modified, all programs, procedures and functions that used the include file must be recompiled. All declarations needed to use the CRSP data with C programs are automatically made by adding the include statements at the beginning of any main programs or subprograms that will use CRSP data or CRSP access or utility routines.

bnd_const.h Include file bnd_const.h contains the constants definitions for programs which access the data in the files.

bnd_struct.h Include file bnd_struct.h contains the structures definitions for programs which access the data in the files.

FILE SPECIFICATIONS

The tables below detail the exact specifications of the formatted CRSP ASCII files. Each table represents one file on the CD. The table names match the names in the CD layout descriptions. The "Character Positions" column shows where in the character record each field is positioned. The "FORTRAN Format" and "C Format" columns are listed in the formats that appear on the CD. The "Associated Name" column refers to the data item defined in the "Description of Definition" section of this guide.

MASTER FILE SPECIFICATIONS

Records are all fixed-length. File names beginning with BX are sorted by QDATE, then by CRSPID. Fields are delimited by a pipe (|).

These files are sorted by CRSPID, then QDATE where available.

HEADER FILE - BMHEADER.DAT

This table details the exact specifications of the formatted Bonds Header File. There is one record for each issue, sorted by CRSPID. This file has a 155-character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 24	A8	%8s	Character	CUSIP
26 - 33	A8	%8s	Character	NAME
35 - 42	I8	%8d	Integer	MATDT
44	I1	%1d	Integer	TYPE
46 - 52	F7.3	%7.3f	Double	COUPRT
54	I1	%1d	Integer	UNIQ
56	I1	%1d	Integer	WHY
58 - 65	I8	%8d	Integer	DATDT
67 - 74	I8	%8d	Integer	BANKDT
76 - 83	I8	%8d	Integer	FCALDT
85 - 90	I6	%6d	Integer	YMCNOT
92	I1	%1d	Integer	NOTICE
94	I1	%1d	Integer	TAX
96	I1	%1d	Integer	FLOWER
98	I1	%1d	Integer	NIPPY
100 - 107	I8	%8d	Integer	FCPDT
109	I1	%1d	Integer	FCPDTF
111 - 119	F9.6	%9.6f	Double	VALFC
121 - 125	I5	%5d	Integer	NUMDBT
127 - 131	I5	%5d	Integer	NUMPAY
133 - 137	I5	%5d	Integer	FSTQUO
139 - 143	I5	%5d	Integer	LSTQUO
145 - 149	I5	%5d	Integer	FSTYLD
151 - 155	I5	%5d	Integer	LSTYLD

QUOTES FILE - BMQUOTES.DAT

This table details the exact specifications of the formatted Quotes File. There is one record for each quote, sorted by CRSPID then QDATE. This file has a 52 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 24	I8	%8d	Integer	QDATE
26 - 36	F11.6	%11.6f	Double	BID
38 - 48	F11.6	%11.6f	Double	ASK
50	A1	%1s	Character	SOURCE

YIELD FILE - BMYIELD.DAT

This table details the exact specifications of the formatted Yields File. There is one record for each quote, sorted by CRSPID then QDATE. This file has a 74 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 24	I8	%8d	Integer	QDATE
26 - 38	E13.6	%13.6E	Double	ACCINT
40 - 52	E13.6	%13.6E	Double	YLD
54 - 66	E13.6	%13.6E	Double	RETNUA
68 - 73	F6.1	%6.1f	Double	DURATN

DEBT FILE - BMDEBT.DAT

This table details the exact specifications of the formatted Debt File. There is one record for each amount outstanding observation sorted by CRSPID then PQDATE. This file has a 38 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 24	I8	%8d	Integer	BQDATE
26 - 31	I6	%6d	Integer	TOTOUT
33 - 36	I6	%6d	Integer	PUBOUT

COUPON PAYMENTS FILE - BMPAYMTS.DAT

This table details the exact specifications of the formatted Bonds Payments File. There is one record for each amount outstanding observation, sorted by CRSPID then DQDATE. This file has a 36 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 24	I8	%8d	Integer	PQDATE
26 - 34	F9.6	%9.6f	Double	PDINT

ADDRESS FILE - BMADDRS.DAT

This table details the exact specifications of the formatted CRSP daily master address file. The CRSP Daily Master Address File contains one record for each issue, and contains header information used by CRSP sample programs to read other Master Files randomly. This file has a 95 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 25	I9	%9d	Integer	DBTLOC
27 - 35	I9	%9d	Integer	DBTSIZ
37 - 45	I9	%9d	Integer	PAYLOC
47 - 55	I9	%9d	Integer	PAYSIZ
57 - 65	I9	%9d	Integer	QUOLOC
67 - 75	I9	%9d	Integer	QUOSIZ
77 - 85	I9	%9d	Integer	YLDLOC
87 - 95	I9	%9d	Integer	YLDSIZ

CROSS-SECTIONAL FILE SPECIFICATIONS

These files are sorted by QDATE, then CRSPID where available.

CALENDAR FILE - BXCALIND.DAT

This table details the exact specifications of the formatted Calendar/Rates File. There is one record for each Quote Date, sorted by QDATE. This file has an 87 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 8	I8	%8d	Integer	QDATE
10 - 17	I8	%8d	Integer	DELDAT
19 - 24	F6.2	%6.2f	Real	CD1M
26 - 31	F6.2	%6.2f	Real	CD3M
33 - 38	F6.2	%6.2f	Real	CD6M
39 - 44	F6.2	%6.2f	Real	CP30D
46 - 51	F6.2	%6.2f	Real	CP60D
53 - 59	F6.2	%6.2f	Real	CP90D
61 - 66	F6.2	%6.2f	Real	FFEFRT
68 - 73	F6.2	%6.2f	Real	FFMINR
75 - 80	F6.2	%6.2f	Real	FFMAXR
82 - 87	I6	%6d	Integer	NUMACT

QUOTES FILE - BXQUOTES.DAT

This table details the exact specifications of the formatted Cross-Sectional Quotes File. There is one record for each quote, sorted by QDATE then CRSPID. This file has a 52 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 15	A15	%15s	Character	CRSPID
17 - 24	I8	%8d	Integer	QDATE
26 - 36	F11.6	%11.6f	Double	BID
38 - 48	F11.6	%11.6f	Double	ASK
50	A1	%1s	Character	SOURCE

YIELD FILE - BXYIELD.DAT

This table details the exact specifications of the formatted Cross-Sectional Yields File. There is one record for each quote, sorted by QDATE then CRSPID. This file has a 74 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1-8	I8	%8d	Integer	CRSPID
10-24	A15	%15s	Character	QDATE
26-38	E13.6	%13.6E	Double	ACCINT
40-52	E13.6	%13.6E	Double	YLD
54-66	E13.6	%13.6E	Double	RETNUA
68-73	F6.1	%6.1f	Double	DURATN

ADDRESS FILE - BXADDRS.DAT

This table details the exact specifications of the formatted Cross-Sectional Address File. There is one record for each issue, and contains header information used by CRSP sample programs to read other Cross-Sectional Files randomly. This file has a 49 character record.

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1 - 8	I8	%8d	Integer	CRSPID
10 - 18	I9	%9d	Integer	QUOLOC
20 - 28	I9	%9d	Integer	QUOSIZ
30 - 38	I9	%9d	Integer	YLDLOC
40 - 48	I9	%9d	Integer	YLDSIZ

FIXED TERM INDICES FILE SPECIFICATIONS

BXDLYIND.DAT

This table details the exact specifications of the formatted Bonds Fixed Term Indices. There is one record for each maturity term type for each quote date. This file has a 102 character record.

Individual files for each separate term type also exist:

- bxdlyind_1yr.dat
- bxdlyind_2yr.dat
- bxdlyind_5yr.dat
- bxdlyind_7yr.dat
- bxdlyind_10yr.dat
- bxdlyind_20yr.dat
- bxdlyind_30yr.dat

CHARACTER POSITIONS	FORTRAN FORMAT	C FORMAT	DATA TYPE	ASSOCIATED NAME
1-4	I4	%4d	Integer	TERMTYPE
6-13	I8	%8d	Integer	QDATE
15-29	A15	%15s	Character	CRSPID
31-36	F6.3	%6.3f	Doublett	YEARSTM
38-48	F11.6	%11.6f	Double	RETADJ
50-60	F11.6	%11.6f	Double	YTM
62-72	F11.6	%11.6f	Double	ACCINT
74-79	F6.1	%6.1f	Double	DURATN
81-90	F10.6	%10.6f	Double	BID
92-101	F10.6	%10.6f	Double	ASK

EXCEL FILES

The Excel Workbook files do not contain the large Master and Cross-Sectional Files. These files are too large to be supported in Excel. The Excel Files were imported from the ASCII files. The number of decimal places matches those in the original ASCII Files. Therefore, adding decimal places in the cell formatting will not improve accuracy in data output. The dates are stored as Excel dates and displayed in a MM/DD/YYYY format, unless otherwise indicated on the readme worksheet. The first worksheet in each file is a readme worksheet that outlines the contents of the rest of the sheets.

The following table contains the file name, the work sheet names within them, and the section of the documentation that describes them.

FILES	WORK SHEET NAMES	DOCUMENTATION REFERENCE
bmheader.xls	BMHEADER.XLS	Section 4.3, File Specifications, Master File Specifications
bxcalind.xls	BXCALIND.XLS	Section 4.3, Cross-Sectional File Specifications
bxdlyind_10yr.xls	BXDLYIND_10YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications
bxdlyind_1yr.xls	BXDLYIND_1YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications
bxdlyind_20yr.xls	BXDLYIND_20YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications
bxdlyind_2yr.xls	BXDLYIND_2YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications
bxdlyind_30yr.xls	BXDLYIND_30YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications
bxdlyind_5yr.xls	BXDLYIND_5YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications
bxdlyind_7yr.xls	BXDLYIND_7YR.XLS	Section 4.3, Daily Fixed Term Indices File Specifications

Microsoft Excel Support Disclaimer

CRSP does not support Microsoft Excel. These files have been included in this format as a courtesy. If you are unable to load the files or to use the software, please contact Microsoft or your System Administrator for support. These files are in ASCII in the \DATA\ directory if you want to convert them yourself.

SAS FILES

The following table lists SAS data sets available in the SAS directory. These data sets can be used with no additional conversions needed.

EXTRACTED FILE NAMES	DOCUMENTATION REFERENCE
BMDEBT.SAS7BDAT	"DEBT - Amounts Outstanding" on page 20 (Master File)
BMDEBT.SAS7BNDX	CRSPID index for the BMDEBT File
BMHEADER.SAS7BDAT	"HEADER - Issue Identification, Characteristics, and Data Ranges" on page 14 (Master File)
BMHEADER.SAS7BNDX	CRSPID index for the BMHEADER File
BMPAYMTS.SAS7BDAT	"PAYMENTS - Interest Payments" on page 20 (Master File)
BMPAYMTS.SAS7BNDX	CRSPID index for the BMPAYMTS File
BMQUOTES.SAS7BDAT	"QUOTES - Raw Data" on page 17 (Master File)
BMQUOTES.SAS7BNDX	CRSPID index for the BMQUOTES File
BMYIELD.SAS7BDAT	"YIELDS - Derived Data" on page 18 (Master File)
BMYIELD.SAS7BNDX	CRSPID index for the BMYIELD File
BXCALIND.SAS7BDAT	"CALENDAR - Calendar and Government Rates" on page 13 (Cross Sectional File)
BXDLYIND.SAS7BDAT	"CRSP Fixed Term Indices Files" on page 21 (Cross Sectional File)
BXQUOTES.SAS7BDAT	"QUOTES - Raw Data" on page 17 (Cross Sectional File)
BXQUOTES.SAS7BNDX	QDATE index for BXQUOTES
BXYIELD.SAS7BDAT	"YIELDS - Derived Data" on page 18 (Cross Sectional File)
BXYIELD.SAS7BNDX	QDATE index for BXYIELD

SAS Support Disclaimer

CRSP does not support SAS. These files have been included in this format as a courtesy. If you are unable to load the files or to use the software, please contact SAS or your System Administrator for support. The files are in ASCII in the \DATA\ directory if you want to convert them yourself.

APPENDIX A: SPECIAL ISSUES

ISSUES WITH SPECIAL PROVISIONS

The following is a list of issues having special provisions and coded with ITYPE = 9. You may wish to consider these provisions before using the data from these issues.

19330315.902000	Redeemable at option of holder at par plus accrued interest with 60 days notice. Principal and interest payable in United States gold coin.
19340415.904250	Issue created by early call of 19381015.904250. Similar numbers selected to be called for redemption on 19340415 were promulgated by the Treasury effectively creating a new issue which was quoted separately up to the call date.
19341015.904250	Issue created by early call of 19381015.904250. Similar to 19340415.904250.
19350415.904250	Issue related by early call of 19381015.904250. Similar to 19340415.904250.
19381015.904250	Principal and interest payable in United States gold coin.
19451015.903250	Accrued interest at the rate of 4¼% up to 19341015 and at 3¼% thereafter.
19590801.904000	Issue created from 19610801.904000 (see below).
19600215.904000	Issue created from 19620815.904000 (see below).
19610801.904000	Redeemable at the option of the holder at par and accrued interest on August 1, 1959. Notice of intent to redeem must be made by May 1, 1959 and certificates to be redeemed to be stamped. Once stamped, certificates mature on August 1, 1959 (not August 1, 1961 as issued). These stamped certificates were traded and quoted under the new CRSPID, even though no such security was actually issued by the treasury.
19620815.904000	Similar to 19600801.90400. Redeemable at option of holder on February 15, 1960, written notice and surrender required on or before November 16, 1959. Issue thus created was 19600215.904000.
99990401.902000	Consol bond, paid interest quarterly in perpetuity. Principal returned only if called. Issue actually called in 1935.

STRIPPED NOTES AND BONDS

Stripped notes and bonds are issues that have been broken into their respective component cash flows, each of which is then traded separately. This was originally done by various financial institutions who issued treasury backed securities (e.g., CATS, TIGERS etc.). A fully-constituted Treasury note or bond consists of a principal payment and semiannual interest payments. In 1985, the treasury began participating in this market by designating certain issues as eligible to be stripped. All 10-year notes and all bonds issued since November 15, 1984 have been made eligible for the STRIPS program either upon their original issue or after their first interest payment date. Issues so designated could be broken up and the individual cash flows registered separately. As of September 1997, all new Treasury marketable fixed-rate notes and bonds issued on or after September 30, 1997, are eligible for STRIPS. The Treasury itself did not sell the individual payments, this being done by dealers who first purchased eligible securities.

The following issues have been designated as eligible for stripping by the Treasury:

19941115.211620	20000815.208750	20050815.206500	20200815.108750
19950215.211250	20001115.205750	20051115.205870	20210215.107870
19950515.211250	20001115.208500	20060215.109370	20210515.108120
19950815.210500	20010215.207750	20060515.206870	20210815.108120
19951115.209500	20010515.208000	20060715.207000	20211115.108000
19960215.208870	20010815.207870	20061015.206500	20220815.107250
19960515.207370	20011115.207500	20060215.205620	20221115.107620
19961115.207250	20011115.208500	20070215.206250	20230215.107120

19970515.208500	20020515.207500	20070515.206620	20230815.106250
19970815.208620	20020815.206370	20070815.206120	20241115.107500
19971115.208870	20020930.205870	20141115.511750	20250215.107620
19980215.208120	20021031.205750	20150215.111250	20250815.106870
19980515.209000	20021130.205750	20150815.110620	20260215.106000
19980815.209250	20021231.205620	20151115.109870	20260815.106750
19981115.208870	20030215.206250	20160215.109250	20261115.106500
19990215.208870	20030815.205750	20160515.107250	20270215.106620
19990515.209120	20040215.205870	20161115.107500	20270815.106370
19990815.208000	20040515.207250	20170515.108750	20271115.106120
19990930.205750	20040815.207250	20170815.108870	20280815.105500
19991031.205620	20041115.111620	20180515.109120	20281115.105250
19991115.207870	20041115.207870	20181115.109000	20290215.105250
19991130.205620	20050215.207500	20190215.108870	20290815.106120
19991231.205620	20050515.112000	20190815.108120	
20000215.208500	20050515.206500	20200215.108500	
20000515.208870	20050815.110750	20200515.108750	

These issues are also traded as normal notes and bonds and are quoted as such in the files.

FOREIGN TARGETED SECURITIES

Foreign targeted issues are not included in the CRSP US Treasury Database. Certain recent notes have been issued in pairs with identical coupon rates, maturities and dated dates. One issue of the pair is intended for domestic holders and is normal in all respects. The other issue is intended for United States aliens. These “Foreign Targeted Securities” are exempt from certain federal taxes when held by eligible foreigners. They pay interest annually and may be converted into their domestic equivalent or sale to domestic holders. The converse is not true.

The following notes which are included are known to have Foreign Targeted equivalents:

19880930.211370	dated 19841031
19900215.211000	dated 19841203
19900815.209870	dated 19850604
19960215.208870	dated 19860215